Universidad Politécnica
de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Grado en Ingeniería Informática

Trabajo Fin de Grado

Implementación de un modelo de extracción de relaciones semánticas basado en modelos de lenguaje para el español.

Autor: Gonzalo de la Rosa Palacio
Tutor: Elena Montiel Ponsoda
Cotutor: Pablo Calleja Ibáñez

Madrid, julio - 2023

Este Trabajo Fin de Grado se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Grado
Grado en Ingeniería Informática

Título: Implementación de un modelo de extracción de relaciones semánticas basado en modelos de lenguaje para el español.

Julio - 2023

Autor: Gonzalo de la Rosa Palacio
Tutor: Elena Montiel Ponsoda
Cotutor: Pablo Calleja Ibáñez
Departamento de Lingüística Aplicada a la Ciencia y a la Tecnología
Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid

Índice general

Índice de Figuras

Resumen

La identificación de relaciones semánticas es una tarea de procesamiento de lenguaje natural que pretende detectar en un texto el tipo de relación que existe entre dos entidades.

Esta tarea suele consistir en la identificación de relaciones entre entidades previamente clasificadas en grupos o clases. Tradicionalmente, esta tarea se ha centrado en la clasificación de entidades en grupos genéricos como, por ejemplo, persona, organización o localización, y la identificación de la relación que existe entre ellas, que puede ser del tipo: vive en, casado con, parte de, empleado en, etc.

Por otro lado, los modelos de lenguaje basados en Transformers son modelos de aprendizaje profundo generados para representar el conocimiento de un idioma. Estos modelos tienen la particularidad de que pueden ser adaptados y reentrenados para diferentes tareas como la extracción de relaciones mediante conjuntos de datos específicos que representen estas relaciones sin partir de cero, es decir, utilizando todo el conocimiento aprendido previamente en el modelo de lenguaje.

En este trabajo se implementará un modelo de extracción de relaciones semánticas en español, ya que, en contraposición al inglés, existe poco desarrollo de modelos específicos para el primer idioma.

Aprovechando el potencial que han demostrado tener los modelos basados en Transformers para tareas relacionadas con el procesamiento de lenguaje natural, utilizaremos un gran modelo de lenguaje preentrenado multilingüe, llamado BART, al que se realizará un ajuste fino sobre un *dataset* creado en español para esta tarea. Seguiremos los pasos de REBEL (*Relation Extraction By End-to-end Language generation*) [1] por la capacidad que ha evidenciado en obtener resultados estado-del-arte con una aproximación innovadora.

Palabras Clave: extracción de relaciones, semántica, procesamiento de lenguaje natural, texto, entidades, clasificación, modelo de lenguaje, Transformers, aprendizaje profundo, español, REBEL, linearización, seq2seq, aprendizaje supervisado, ajuste fino, BART.

Abstract

Semantic relation identification is a natural language processing task that aims to detect in a text the type of relation that exists between two entities.

This task usually consists of identifying relationships between entities previously classified into groups or classes. Traditionally, this task has focused on the classification of entities into generic groups such as person, organisation or location, and the identification of the relationship that exists between them, which can be of the type: lives in, married to, part of, employed in, etc.

On the other hand, Transformer-based language models are deep learning models generated to represent the knowledge of a language. These models have the particularity that they can be adapted and retrained for different tasks such as the extraction of relations by means of specific datasets that represent these relations without starting from scratch, i.e. using all the knowledge previously learned in the language model.

In this work we will implement a semantic relation extraction model in Spanish, since in contrast to English, there is little development of specific models for the first language.

Taking advantage of the potential that Transformer-based models have demonstrated for tasks related to natural language processing, we will use a large multilingual pretrained language model, called BART, which will be fine-tuned on a dataset created in Spanish for this task. We will follow in the footsteps of REBEL (Relation Extraction By End-to-end Language generation) [1] because of its ability to obtain state-of-the-art results with an innovative approach.

Keywords: relation extraction, semantics, natural language processing, text, entities, classification, language model, Transformers, deep learning, Spanish, REBEL, linearisation, seq2seq, supervised learning, fine tuning, BART.

Capítulo 1

Introducción

El procesamiento de lenguaje natural (PLN) es una rama de la inteligencia artificial que se encarga de la interacción entre los ordenadores y los seres humanos a través del lenguaje natural. El lenguaje natural es aquel que utilizamos los seres humanos para comunicarnos de forma normal, a través de las llamadas lenguas naturales, como por ejemplo, el inglés o el español. El objetivo es dotar a las máquinas de la capacidad de entender, interpretar y generar el lenguaje humano, de manera que puedan procesar y analizar grandes cantidades de datos de tipo textual de forma eficiente y precisa.

El PLN se ha vuelto cada vez más importante en la era digital, en la que se generan enormes cantidades de datos, incluyendo los de tipo textual en diferentes idiomas y formatos, como correos electrónicos, publicaciones en redes sociales, documentos de texto, entre otros. Estos datos son una fuente valiosa de información, pero su análisis puede resultar difícil y costoso para los seres humanos. Aquí es donde el PLN juega un papel fundamental, permitiendo el análisis y procesamiento automatizado de grandes cantidades de texto en diferentes idiomas.

Entre las aplicaciones más comunes del PLN se encuentran los asistentes virtuales, sistemas de traducción automática, análisis de sentimientos, resumen de textos, entre otros. Además, el PLN está siendo utilizado en campos como la medicina, el derecho y la educación, donde se están desarrollando sistemas que ayudan a los profesionales a analizar grandes cantidades de información de manera más rápida y precisa.

Es bien conocido el impacto que estas herramientas están empezando a tener sobre la población general, como es el caso de ChatGPT, el chatbot desarrollado por la compañía OpenAI que ha tenido una adopción masiva a escala global. Esta tecnología ha sido la de más rápido crecimiento de la historia, en términos de usuarios activos[1].

En resumen, el PLN es una disciplina en constante evolución que tiene como objetivo facilitar la interacción entre humanos y máquinas en lenguaje natural.

El PLN se puede subdividir en diferentes tareas de investigación que cubren diferentes aspectos del lenguaje. Este TFG se centra particularmente en la tarea de identificación de relaciones semánticas.

[1] https://time.com/6253615/chatgpt-fastest-growing/

La identificación de relaciones semánticas es una tarea clave dentro del PLN que consiste en detectar el tipo de relación existente entre dos entidades dentro de un texto. Las entidades son los nombres o términos que pueden ser identificados y clasificados en grupos o clases. Estas clasificaciones de entidades suelen ser en personas, organizaciones, lugares, productos, fechas, entre otros. La identificación de la relación entre las entidades viene determinada por la interacción que dos entidades tienen entre sí. Por ejemplo «parte de», «vive en», «casado con», «empleado en», etc. Estas relaciones suelen venir dadas por verbos o sintagmas verbales con el sujeto de la oración.

Esta es una tarea fundamental y su importancia radica en que proporciona una representación estructurada del conocimiento contenido en los textos, lo que es esencial para una amplia variedad de aplicaciones de procesamiento de lenguaje natural, como respuesta a preguntas, la recuperación de información y la traducción automática.

En este trabajo, el objetivo es implementar un modelo de extracción de relaciones semánticas en español. Si bien hay una gran cantidad de trabajos y modelos previos en inglés, la investigación en español es limitada en comparación con este y otros idiomas. Este enfoque busca llenar esta brecha en la investigación en el campo de la extracción de relaciones semánticas en español, con el objetivo de avanzar en la comprensión y aplicación de las relaciones semánticas en este idioma en diferentes dominios y mejorar la representación estructurada del conocimiento en texto en español.

Este trabajo se enmarca en el paradigma de aprendizaje supervisado, donde el modelo será entrenado utilizando el conjunto de datos creados, que consta de frases «crudas» y su respectiva linearización con las relaciones semánticas entre las entidades presentes en cada frase. Por linearización se entiende el proceso de codificar una relación semántica en forma de texto siguiendo una estructura definida a priori muy concreta. Por ejemplo: «Juan trabaja en Accisan» se podría linearizar en «Juan <suj>Accisan<obj>trabaja en», donde <suj> y <obj> indican el sujeto y objeto de la relación respectivamente. Otra terminología usada para esto es entidades «cabeza» (*head*) y «cola» (*tail*).

Para llevar a cabo nuestro objetivo, aprovecharemos el poder y la eficacia demostrados por los modelos de lenguaje basados en Transformers para tareas relacionadas con el PLN. Los Transformers son una arquitectura de red neuronal que ha demostrado un gran éxito en tareas como la traducción automática y el procesamiento de lenguaje natural en general.

En particular, utilizaremos un gran modelo de lenguaje preentrenado: BART, que es multilingüe y tiene un gran rendimiento en una amplia variedad de tareas. Realizaremos un ajuste fino a este modelo sobre un conjunto de datos en español que recopilaremos específicamente para este propósito. El ajuste fino es un método de aprendizaje por transferencia en el que los pesos de un modelo preentrenado se entrenan con datos nuevos.

Para la extracción de relaciones semánticas, en el enfoque tradicional, el problema se aborda con dos pasos sucesivos bien diferenciados. Primero, el REN (Reconocimiento de Entidades Nombradas), en donde se identifican y marcan las entidades del texto; y segundo, la CR (Clasificación de la Relación), en que se especifica el tipo de relación entre cada par de las entidades previamente reconocidas.

Nosotros, por el contrario, seguiremos una aproximación distinta y reformularemos la extracción de relaciones como una tarea seq2seq. Seq2seq utiliza una red neuronal recurrente para traducir una secuencia de texto en una secuencia de etiquetas de relación. Este enfoque integrado es más eficiente y efectivo en la extracción de relaciones semánticas. En particular, seguiremos la metodología de REBEL (*Relation Extraction By End-to-end Language generation*) [1], que se centra en generar directamente la relación semántica entre las entidades a partir del texto bruto.

En nuestro trabajo por tanto se abordará la construcción de una herramienta de extracción de relaciones semánticas en español, que permita a los investigadores y desarrolladores utilizar nuestro modelo para extraer relaciones semánticas para una amplia variedad de aplicaciones.

Además, evaluaremos la efectividad de nuestro modelo mediante una serie de experimentos, y compararemos su rendimiento con otros modelos de extracción de relaciones semánticas existentes.

El objetivo principal del trabajo es crear recursos para el desarrollo de la extracción de relaciones en español, de forma que estos queden accesibles públicamente para la disposición de investigadores y tecnólogos que puedan aprovecharlos. Este objetivo se desgrana en tres:

- Identificar conjuntos de datos en español que estén anotados con relaciones
- Crear un corpus actualizados de extracción de relaciones para el español siguiendo las nuevas metodologías del estado de la cuestión
- Usar modelos de lenguaje preentrenados en español y reajustarlos para la tarea de extracción de relaciones

Esta memoria del Trabajo de Fin de Grado está estructurada en 7 capítulos. El capítulo 1 de introducción, el presente. El capítulo 2 muestra el trabajo relacionado y el estado de la cuestión. El capítulo 3, el desarrollo. El capítulo 4 contiene los resultados. El capítulo 5 cubre el impacto del trabajo.

Capítulo 2

Trabajo relacionado y Estado de la Cuestión

La investigación sobre la extracción de relaciones semánticas ha sido un tema ampliamente estudiado en el campo del procesamiento del lenguaje natural. En este capítulo haremos un breve análisis histórico de la ER con 3 enfoques: primeramente veremos los sucesivos paradigmas o fases que ha habido en esta tarea desde su concepción; seguidamente, examinaremos el estado del arte y las técnicas más recientes utilizadas; y finalmente, nos enfocaremos específicamente en el desarrollo de la tarea en el idioma español.

Antes de comenzar, abordaremos la terminología que utilizaremos. El término «extracción de relaciones» (*relation extraction* en inglés) ha sido utilizado con diferentes concepciones o connotaciones [2]. Por lo tanto, utilizaremos el concepto de ER (Extracción de Relaciones) con un significado específico de «extraer las relaciones semánticas entre entidades a partir del texto sin etiquetas que delimiten las entidades»; también conocido como ER de extremo a extremo. Por otro lado, utilizaremos CR (Clasificación de Relaciones) con el significado de «clasificar la relación semántica entre un par específico de entidades dadas en un contexto determinado».

Siguiendo el criterio de Hogan [3], podemos clasificar la historia de los métodos de ER en cuatro grandes fases: (1) basados en patrones, (2) basados en estadísticas, (3) basados en redes neuronales y (4) basados en grandes modelos de lenguaje.

En la primera fase, basada en patrones, que comenzó aproximadamente en 1970, se desarrollaron algoritmos para aprender patrones lingüísticos y extraer información de relaciones de texto. Estos métodos utilizan palabras o frases semilla combinadas con fuentes de conocimiento, como etiquetadores de partes del discurso y clasificación semántica, para identificar patrones relevantes en el texto.

En la segunda fase, basada en estadísticas, que comenzó alrededor de 2004, se utilizaron métodos avanzados de aprendizaje automático estadístico para extraer relaciones de texto. Estos métodos requerían conjuntos de datos grandes para entrenamiento y evaluación, lo que llevaba a problemas de escasez de datos etiquetados. Para abordar este problema, se introdujo el concepto de supervisión a distancia, que permite generar automáticamente grandes cantidades de datos de entrenamiento utilizando grafos de conocimiento acompañantes. Para generar nuestro *dataset* utiliza-

mos esta aproximación de supervisión a distancia, que explicaremos en detalle más adelante. Se utilizaban herramientas o anotaciones textuales hechas a mano, como árboles de dependencia y etiquetas de partes del discurso, típicamente para entrenar clasificadores de regresión logística.

La tercera fase, basada en redes neuronales, comenzó alrededor de 2010. Aquí, en lugar de la regresión logística, se utilizaron redes neuronales, y en la representación vectorial de los *tokens* se reemplazaron las características explícitas por características implícitas mediante redes neuronales. Hablamos de tecnologías como Word2Vec [4] y GloVe [5]. Esto llevó a mejoras significativas en el rendimiento de los modelos de ER.

La cuarta y última fase hasta la fecha se basa en grandes modelos de lenguaje, como BERT (*Bidirectional Encoder Representations from Transformers*), que se entrenan con un enfoque de dos pasos: preentrenamiento de gran escala y ajuste fino supervisado. Estos modelos preentrenados han mejorado el rendimiento de los modelos de ER en diferentes evaluaciones y se utilizan ampliamente en la actualidad. El modelo que utilizamos en este trabajo, como ya se ha mencionado, se basa en BART, que es un gran modelo de lenguaje. Más adelante cubriremos en detalle sus características.

A continuación, haremos un repaso más detallado de las técnicas más recientes, correspondientes aproximadamente a la última década.

Los primeros enfoques abordaban la ER como una cadena de procesos o *pipeline*, donde primero se identificaban las entidades presentes en el texto mediante el REN (Reconocimiento de Entidades Nombradas) y luego se clasificaba la relación, o la falta de ella, entre cada par de entidades reconocidas, lo que entendemos como CR. Estas aproximaciones utilizaban RNCs (Redes Neuronales Convolucionales) o LSTMs (*Long Short-Term Memories*) para explotar la semántica a nivel de oración y clasificar las relaciones [6, 7]. Estos enfoques pertenecen a la tercera fase de las ya descritas.

Por otro lado, los enfoques más recientes de extremo a extremo utilizan redes neuronales para clasificar todos los pares de palabras presentes en el texto de entrada [8, 9], utilizando una representación de tabla o llenado de tabla, centrándose en llenar las ranuras de una tabla, que representan las relaciones. En [10] también utilizaron una formulación similar basada en tabla, donde la tabla se codifica explícitamente utilizando un codificador de secuencia de tabla. Estos también se integran en la tercera fase.

Por último, existen sistemas de *pipeline* que abordan ambas partes de la ER: REN y CR, mediante la formación conjunta de componentes que aprovechan la información compartida entre las tareas. En estas configuraciones, las entidades se extraen primero como en REN utilizando las llamadas etiquetas BILOU y luego un clasificador biafín extrae sus relaciones, compartiendo parte de los codificadores para ambas tareas. Estos enfoques comprenden desde LSTMs [8, 11] hasta RNCs [12, 7] y, más recientemente, arquitecturas basadas en Transformer [13], que predicen y codifican explícitamente los intervalos de entidad en lugar del enfoque BILOU utilizado en REN. Igualmente estos enfoques forman parte de la tercera fase.

En los últimos tiempos, los modelos de ER a nivel de oración se han basado en modelos Transformer, como BERT [13, 10] o ALBERT [14, 10], que son grandes modelos del lenguaje. Para abordar la ER a nivel de documento, en [15] han utilizado un enfoque de *pipeline* entrenado conjuntamente en una configuración de multitarea que

aprovecha la resolución de correferencias para operar a nivel de entidades en lugar de menciones. Estos pertenecen ya a la cuarta fase.

Aunque la importancia de la tarea de ER ha sido destacada en el trabajo mencionado, la falta de líneas base consistentes y una definición cohesiva de la tarea ha llevado a discrepancias en el uso de conjuntos de datos y en la forma en que se han evaluado los modelos.

En [2] han explicado los diferentes problemas hasta ahora y han intentado unificar la evaluación de la ER y realizar una comparación justa entre los sistemas. Siguiendo las directrices propuestas por [2] y al igual que hacen en REBEL, nosotros utilizaremos una evaluación estricta. Esto significa que una relación se considerará correcta solo si las formas de superficie de las entidades de cabeza y cola se extraen correctamente (es decir, se superponen completamente con la anotación), así como los tipos de relación y entidad (si están disponibles para el conjunto de datos). Hasta ahora, se ha demostrado que los métodos de *pipeline* y tabla tienen un rendimiento favorable en la ER, pero todavía enfrentan desafíos. A menudo, estos métodos asumen solo un tipo de relación entre cada par de entidades como máximo, y los enfoques de múltiples clases no consideran otras predicciones. Por ejemplo, podrían predecir dos fechas de nacimiento para la misma entidad sujeto, o predecir relaciones que son incompatibles entre sí. Además, estos métodos requieren calcular todas las posibles combinaciones de entidades, lo que puede ser computacionalmente costoso.

Los enfoques seq2seq para ER [16, 17, 18] ofrecen algunas soluciones a estos problemas. Los mecanismos de decodificación pueden producir las mismas entidades varias veces y condicionar la decodificación futura en predicciones anteriores, tratando implícitamente las incompatibles. Sin embargo, como discuten en [17], todavía enfrentan algunos problemas. Los tripletes deben linearizarse en un orden secuencial arbitrario, como el alfabético. Este problema es abordado por [19], que utilizan el aprendizaje por refuerzo para calcular el orden de extracción de los tripletes. Además, los enfoques seq2seq pueden sufrir de sesgo de exposición, ya que durante el entrenamiento, la predicción siempre depende de la salida esperada. En [17], se propone un enfoque de decodificación de árbol que mitiga estos problemas, al tiempo que utiliza un enfoque autorregresivo seq2seq.

Por otro lado, los modelos Transformer seq2seq, como BART [20] o T5 [21], se han utilizado en diversas tareas de NLU, como vinculación de entidades [22], análisis AMR [23], etiquetado semántico de roles [24] o desambiguación de sentidos de palabras [25], al reformularlas como tareas seq2seq. Estos modelos no solo muestran un alto rendimiento, sino que también demuestran la flexibilidad de los modelos seq2seq al no depender de conjuntos de entidades predefinidos, sino del mecanismo de decodificación, que puede extenderse fácilmente a entidades nuevas o no vistas.

Para nuestro modelo, siguiendo el ejemplo de REBEL [1], empleamos un marco codificador-decodificador que puede mitigar algunos de los problemas mencionados anteriormente que enfrenta el enfoque seq2seq para la ER. Aunque aún puede haber sesgo de exposición, el mecanismo de atención permite capturar dependencias a larga distancia, así como prestar atención (o no) a la salida ya decodificada. Además, hemos utilizado la linearización de tripletes creada por los investigadores de REBEL, que tiene un orden consistente que permite al modelo aprovechar tanto la entrada codificada como la salida ya decodificada.

Finalmente, y a pesar de los avances recientes en la extracción de relaciones semánticas, la implementación de modelos para el español sigue siendo un desafío debido a la falta de corpus y conjunto de datos anotados para esta tarea. Aunque existen algunos recursos como el corpus de AnCora [26], estos no están etiquetados específicamente para la extracción de relaciones semánticas.

En el trabajo de *Open Information Extraction for Spanish Language* [27], los autores trabajaron en un modelo capaz de extraer relaciones no pertenecientes a un dominio específico. Su enfoque se basaba en constricciones sintácticas utilizando etiquetas POS. Llegaron a la conclusión de que el rendimiento obtenido era similar al de los modelos que usan el mismo método para el inglés. Aun así, este trabajo está lejos del estado de la cuestión de hoy.

En un trabajo reciente, Fabregat [28] aborda de manera detallada la tarea de extracción de relaciones en el dominio biomédico, tanto en inglés como en español, utilizando técnicas de procesamiento de lenguaje natural y aprendizaje automático. El autor se enfoca en la identificación de menciones a discapacidades y su relación con enfermedades raras. Además, se exploró la sinergia entre el reconocimiento de entidades y la extracción de relaciones, en línea de la relación de extracciones de extremo a extremo.

A pesar de todo, se encuentran disponibles algunos conjuntos de datos multilingües que incluyen al español; sin embargo, estos también presentan problemas. Existe un *dataset* multilingüe llamado SMiLER [29], que ha sido desarrollado asimismo con supervisión a distancia. Aun así, tiene inconvenientes: limita las anotaciones a un triplete por oración, o sea, que de cada oración podrá obtener como máximo un triplete, ignorando otras relaciones que puedan estar presentes; además, no es propicio para entrenar sistemas de ER de extremo a extremo, pues ha sido configurado bajo la concepción de *pipeline*.

Por último, consideramos relevante enseñar que también existen otros *datasets* multilingües y que, aun pudiendo ser buenos para la tarea, no son gratuitos. Por ejemplo, ACE05[1], que incluye el español entre sus idiomas pero para poder usarlo es necesario tener una licencia de pago, que va de 2000$ a 4000$. Además, este solo cubre 6 tipos de relación.

[1] https://catalog.ldc.upenn.edu/LDC2006T06

Capítulo 3

Desarrollo

En este capítulo se detalla el desarrollo realizado para cumplir los objetivos establecidos en la introducción. En primer lugar, en la sección 3.1, se hará una breve introducción acerca de los *datasets* de ER, mostrando algunos corpus de referencia. Luego se explicará el fracaso en la búsqueda de corpus aptos en español, que se alinea con el primer subobjetivo: «Identificar conjuntos de datos en español que estén anotados con relaciones». Finalmente, se desarrollará el fundamento teórico en que se basa nuestro *dataset*.

En la sección 3.2 se explica en detalle cRocoDiLe, que es la tecnología que hemos usado para generar nuestro corpus en español de forma automática. También se explicarán los cambios necesarios que hemos tenido que hacer para su correcto funcionamiento. A continuación, en la sección 3.3, se explicará el fundamento teórico de REBEL, la configuración del entorno y librerías para usarlo, y, de nuevo, los cambios necesarios para su correcto funcionamiento. Por último, en la sección 3.4 se explican los detalles de Magerit, que es el entorno de supercomputación en el que se ha entrenado al modelo.

3.1. Corpus de extracción de relaciones

3.1.1. Introducción

En el contexto del aprendizaje automático y de este trabajo, entendemos como *dataset* o corpus una colección de datos anotados que se utiliza para entrenar y evaluar los modelos. En nuestro caso, como queremos que el modelo sea capaz de hacer la tarea de ER, el corpus deberá tener ejemplares o instancias que hagan esto mismo. O sea, cada elemento del corpus estará compuesto de oraciones o fragmentos de texto, y de sus equivalentes entidades involucradas y relaciones que existen entre ellas.

La estructura y el formato exacto de cada *dataset* de ER puede variar según el propósito específico y el diseño del conjunto de datos. Sin embargo, a menudo siguen una estructura similar, en la que como ya se ha dicho se especifican las palabras crudas, sus entidades y relaciones, y a veces también otros datos. Incluso la extensión y codificación de cada corpus puede ser diferente, pero normalmente, para la ER se suelen utilizar ficheros de tipología JSON (*JavaScript Object Notation*). Estos archivos tienen una estructura simple basada en pares clave-valor. Los pares clave-valor consisten

en una clave seguida de dos puntos y un valor asociado. Los valores pueden ser únicos o vectores. Un vector no es más que una lista ordenada de valores. Un ejemplo de este par clave-valor podría ser: «ïd": "182"»

A modo de ejemplo, vamos a examinar una instancia de un famoso corpus, llamado NYT (*New York Times*), que luego presentaremos junto a otros *datasets* de referencia. La estructura del corpus (claves) es la siguiente:

- **tokens**: Una lista de tokens que representan las palabras individuales y crudas del texto.
- **spo_list**: Una lista de listas que contiene información sobre las relaciones entre las entidades. Cada sublista tiene tres elementos: la entidad sujeto, la relación y la entidad objeto.
- **spo_details**: Una lista de listas que proporciona detalles adicionales sobre las relaciones entre las entidades. Cada sublista contiene información sobre los índices de los tokens relacionados y los tipos de entidades involucradas.
- **pos_tags**: Una lista de etiquetas de partes del discurso que representan la categoría gramatical de cada token.

A continuación, una instancia (valores) del mismo:

- **tokens**: [“Demolition”, “has”, “begun”, “on”, “another”, “purchase”, “in”, “Long”, “Island”, “City”, “,”, “Queens”, “.”]
- **spo_list**: [[“Queens”, “/location/location/contains”, “Long Island City”], [“Long Island City”, “/location/neighborhood/neighborhood_of”, “Queens”]]
- **spo_details**: [[11, 12, “LOCATION”, “/location/location/contains”, 7, 10, “LOCATION”], [7, 10, “LOCATION”, “/location/neighborhood/neighborhood_of”, 11, 12, “LOCATION”]]
- **pos_tags**: [“NN”, “VBZ”, “VBN”, “IN”, “DT”, “NN”, “IN”, “NNP”, “NNP”, “NNP”, “,”, “NNP”, “.”]

Como ya se ha adelantado, existen unos corpus anotados y validados manualmente que se utilizan como marcos de referencia (*benchmarks* en inglés) y que son utilizados por la comunidad para hacer comparaciones de los resultados obtenidos. Es decir, utilizan estos *datasets* para entrenar sus modelos y así poder evaluar el rendimiento de los modelos de forma coherente con el de los demás.
Destacan DOCRED, TACRED, SemEval-2010 Task 8, NYT o CONLL04. Podemos consultarlos en el sitio web *Papers with code*[1].

CONLL04 (*Conference on Computational Natural Language Learning*) [30] está compuesto por oraciones de artículos de noticias, anotadas con 4 tipos de entidades (persona, organización, ubicación y otro) y 5 tipos de relaciones (mata, trabaja para, organización con sede en, vive en y ubicado en).

DocRED (*Document-Level Relation Extraction Dataset*) [31] es un conjunto de datos reciente creado con una metodología parecida a cRocoDiLe, aprovechando Wikipedia y Wikidata. Sin embargo, se centra en fragmentos de texto más largos, con relaciones

[1] https://paperswithcode.com/task/relation-extraction

entre entidades a nivel de documento. Incluye anotaciones para 6 tipos de entidades diferentes y 96 tipos de relaciones.

NYT (*New York Times*) [32], el ya ejemplificado, es un conjunto de datos que consiste en oraciones de noticias del corpus del homónimo periódico. El *dataset* contiene relaciones anotadas de forma remota utilizando FreeBase[2]. Concretamente, nos fijamos en la versión procesada de Zeng et al. (2018) [16] llamada NYT-multi, que contiene entidades superpuestas, con 3 tipos de entidades diferentes y 24 tipos de relaciones.

TACRED (*The TAC Relation Extraction Dataset*) [7] es un conjunto de datos de ER a gran escala construidos a partir de textos de agencias de noticias y páginas web del corpus utilizado en los desafíos anuales de TAC KPB (*TAC Knowledge Base Population*). Las instancias de las que se compone se crean combinando anotaciones humanas disponibles de los desafíos TAC KBP y *crowdsourcing*. Estas cubren 41 tipos de relaciones o se etiquetan como «no relación» si no se mantiene ninguna relación definida, a diferencia de los otros corpus, las entidades están codificadas de manera indirecta en las propias relaciones en vez de explícitamente como elementos autónomos.

3.1.2. Identificación de corpus de ER en español

Tanto estos corpus de referencia de los que hemos hablado, como aquellos que han quedado fuera, no están en español, que es lo que necesitamos para nuestro modelo. Después de haber hecho una búsqueda intensiva no se ha encontrado ninguno que supere los criterios de calidad para esta desarrollar esta tarea en español. Los criterios a los que nos referimos son los siguientes:

- **Accesibilidad**: criterio inicial y necesario, que sea un corpus no privado y accesible al público general, o en el peor de los casos, a los investigadores.
- **Gratuidad**: además de ser accesible, buscamos *datasets* gratuitos, de forma que puedan ser utilizados por todo aquel que lo considere, con independencia de su poder adquisitivo.
- **Dominio**: hay *datasets* que son de un dominio específico, por ejemplo el biomédico, limitando su uso a textos de esta naturaleza.
- **Tamaño**: referente a la cantidad de instancias contenidas en el corpus, pues para entrenar a los grandes modelos de lenguaje, hace falta mucha cantidad de datos.
- **Cantidad de relaciones**: ya que queremos que el modelo sea lo más flexible posible, de forma que sea capaz de conocer un buen número de relaciones diferentes.
- **Especificidad de tarea**: nosotros buscamos un corpus que esté pensado para la ER de extremo a extremo, pero muchos están concebidos para usarse de otra forma, como para reconocimiento de entidades nombradas seguido de clasificación de relación.
- **Otras limitaciones**: algunos *datasets* tienen limitaciones específicas, como el ya citado SMiLER [29], que a lo sumo reconoce un triplete por oración. Nosotros queremos evitar este tipo de constricción.

[2]https://es.wikipedia.org/wiki/Freebase

Esta situación es la que nos da la motivación de crear un *dataset* propio, que a priori pasa estos criterios de calidad. Como ya se ha mencionado, los detalles acerca de su generación se encuentran en la sección de 3.2.

3.1.3. Creación de un corpus propio

En nuestro caso, el corpus que utilizaremos para entrenar al modelo es lo que se conoce como «dataset plateado» (*silver dataset*), que se refiere a un conjunto de datos generados de forma automática, tal y como se explica de forma pragmática y detallada en el apartado de 3.2. Este tipo de corpus tiene como contrapartida el llamado «dataset dorado» (*gold dataset*), que es etiquetado manualmente y para el que se supone una mayor precisión en la anotación, y por ende, calidad.

Cabría preguntarse entonces el porqué de utilizar un *dataset* plateado en vez de uno dorado, pero la respuesta es evidente. El problema de los *datasets* dorados es que generarlos es tanto caro como lento. Hay que contratar a trabajadores que hagan la labor de etiquetado, y, teniendo en cuenta el tamaño de los *datasets* que necesitamos, eso se traduce en una gran inversión económica. Además, estos etiquetadores no dejan de ser humanos, por lo que el trabajo que desempeñan no es inmediato, y el tiempo que se necesita para generar los *datasets* a veces es demasiado grande.

Este corpus plateado va a ser generado a través de una metodología llamada «supervisión a distancia», término acuñado en [33]. Para esto, se utiliza un grafo de conocimiento, que es un grafo de entidades conectadas por pares a través de aristas etiquetadas con el tipo de relación que guardan esas entidades. Entonces, la supervisión a distancia para la ER es un método que combina un grafo de conocimiento con un corpus no estructurado (texto crudo) para generar datos etiquetados automáticamente.

Primero, se identifican las entidades del grafo de conocimiento que también estén presentes en el texto, y luego se hace la siguiente suposición: todo par de entidades presentes en las oraciones expresan el mismo tipo de relación que está especificada en el grafo de conocimiento. La Figura 3.1 muestra un ejemplo de etiquetas de relación generadas automáticamente mediante la combinación de un grafo de conocimiento (en este caso de Wikidata) con oraciones crudas. A partir de aquí, es trivial obtener los tripletes.

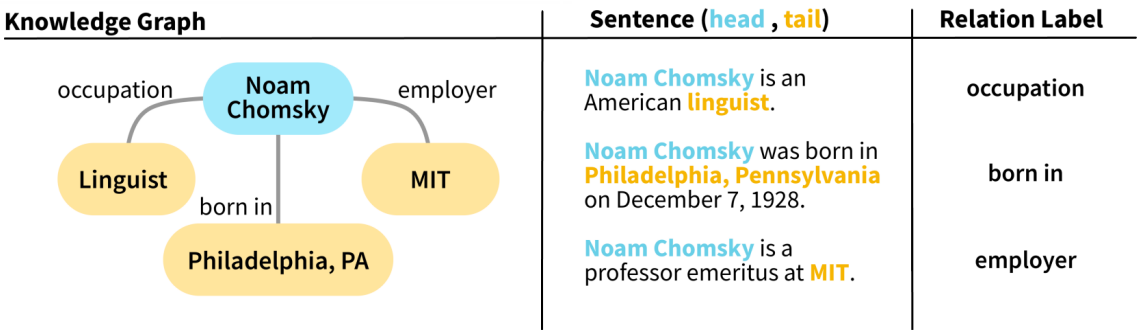

Figura 3.1: Etiquetas de relación generadas por la supervisión a distancia.

Otra particularidad de nuestro corpus es que considera la ER a nivel de documento (*document-level RE*), en contraposición a aquellos que lo hacen a nivel de frase (*sentence-level*). Ignorando posibles errores creados por este cambio de «paradigma», este tipo de aproximación es superior en términos de completitud. Esta superioridad viene dada porque es más flexible, o sea, no solo es capaz de encontrar relaciones en la misma frase, sino también entre varias distintas; pudiendo obtener información de relaciones que efectivamente existen y que de otra manera no se hubieran recogido. Esto hace a nuestro trabajo contemporáneo, ya que la ER a nivel de documento se ha convertido en el principal objetivo de los métodos actuales en este campo [3].

3.2. cRocoDiLe

Los modelos transformers autorregresivos, como BART, han demostrado tener un desempeño destacado en diversas tareas de generación, como la traducción o la síntesis de resúmenes. No obstante, es necesario contar con una gran cantidad de datos para poder entrenarlos de manera efectiva. Sin embargo, los *datasets* de ER de extremo a extremo son escasos, normalmente pequeños y casi siempre en inglés. Es en esta situación en la que cRocoDiLe será la llave que nos permita pasar este cuello de botella.

cRocoDiLe, acrónimo de «Automati**c** **R**elati**o**n Extra**c**ti**o**n **D**ataset w**i**th N**L**I Filt**e**ring» es una herramienta para crear un *dataset* para ER de forma automática. También fue lanzado al público en el *paper* de REBEL [1], ya que esta herramienta fue la que ellos mismos utilizaron para generar el *dataset* también bautizado como REBEL. Está disponible en un repositorio GitHub[3], al que hemos accedido para probar y hacer las modificaciones oportunas.

3.2.1. Flujo de funcionamiento

Primero se descargan los ficheros de volcado (*dump* en inglés) de Wikipedia. Un fichero de volcado es un archivo que contiene una copia completa o parcial de la base de datos de Wikipedia en un momento específico. Este fichero de volcado incluye todos los artículos, historiales de revisiones, discusiones, metadatos y otros elementos asociados con el contenido de Wikipedia.

A continuación, se extraen los resúmenes de cada artículo, o sea, el texto que precede inmediatamente a la tabla de contenidos. Entonces, se seleccionan todas las entidades con hipervínculo (o sea, aquellas que tienen un artículo de Wikipedia), incluyendo fechas y valores, y se vinculan con sus artículos correspondientes de Wikidata haciendo uso de Wikimapper.

Wikidata[4] es una base de conocimientos editada en colaboración y alojada por la Fundación Wikimedia, al igual que Wikipedia. Toda entrada de Wikipedia tiene su equivalente en Wikidata, la diferencia es el contenido de esta última, pues estas entradas se basan en las llamadas «declaraciones», que no son más que una lista de relaciones en forma de propiedad-valor (puede haber pares que tengan la misma clave). Por ejemplo, para la entrada «Douglas Adams» encontramos las declaraciones «alma mater: Saint John's College» y «alma mater: Brentwood School», como se ve en

[3] https://github.com/Babelscape/crocodile
[4] https://dl.acm.org/doi/pdf/10.1145/2629489

3.2. En definitiva, cada página de Wikidata almacena una serie de relaciones entre la entidad referente al artículo y otras. Es precisamente por este motivo que Wikidata es una herramienta tan útil para la generación de un dataset de ER, que aprovecharemos con los pasos sucesivos que se están explicando.

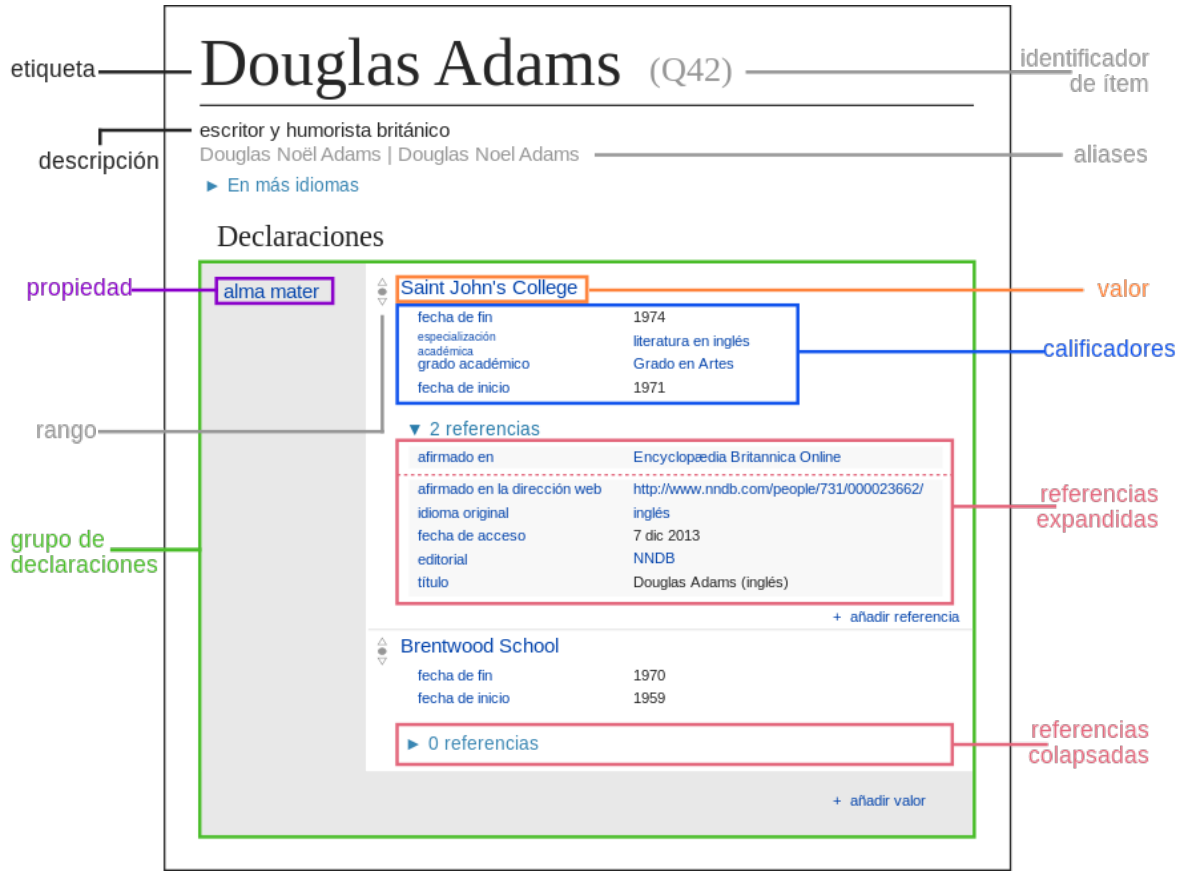

Figura 3.2: Artículo de Douglas Adams en Wikidata con anotaciones.

Acto seguido, se extraen todas las relaciones encontradas entre la entidad sujeto (la que da nombre al artículo) y las otras entidades que se han obtenido gracias al hipervínculo. Estas relaciones se codifican en forma de tripletes. Un triplete es un elemento consistente en un sujeto, un objeto y la relación que los une. Un posible ejemplo sería «Juan<suj>Accisan<obj>trabaja en».

Sin embargo, siguiendo el enfoque de «supervisión a distancia», hemos hecho la asunción de que estas relaciones conseguidas de Wikidata se pueden deducir del texto, lo cual no es necesariamente cierto. Por eso, y en pos de mejorar la precisión y calidad del *dataset*, se hará un filtrado para elegir los tripletes finales. Utilizaremos un modelo preentrenado de RoBERTa para la inferencia de lenguaje natural, que eliminará las relaciones que no se deduzcan de los resúmenes de Wikipedia. Para cada relación que haya pasado el filtro, se introducirá el texto crudo que contiene ambas entidades relacionadas y el triplete asociado, separados por el token <sep>. Se mantendrán únicamente aquellos tripletes cuya predicción de vinculación sea superior a 0,75.

Finalmente, se crearán tres divisiones aleatorias del conjunto de datos resultante, donde la validación y evaluación representan el 5 % del total de los datos cada una.

O sea, un 90% de los datos para entrenamiento, un 5% para la validación y el 5% restante para la evaluación.

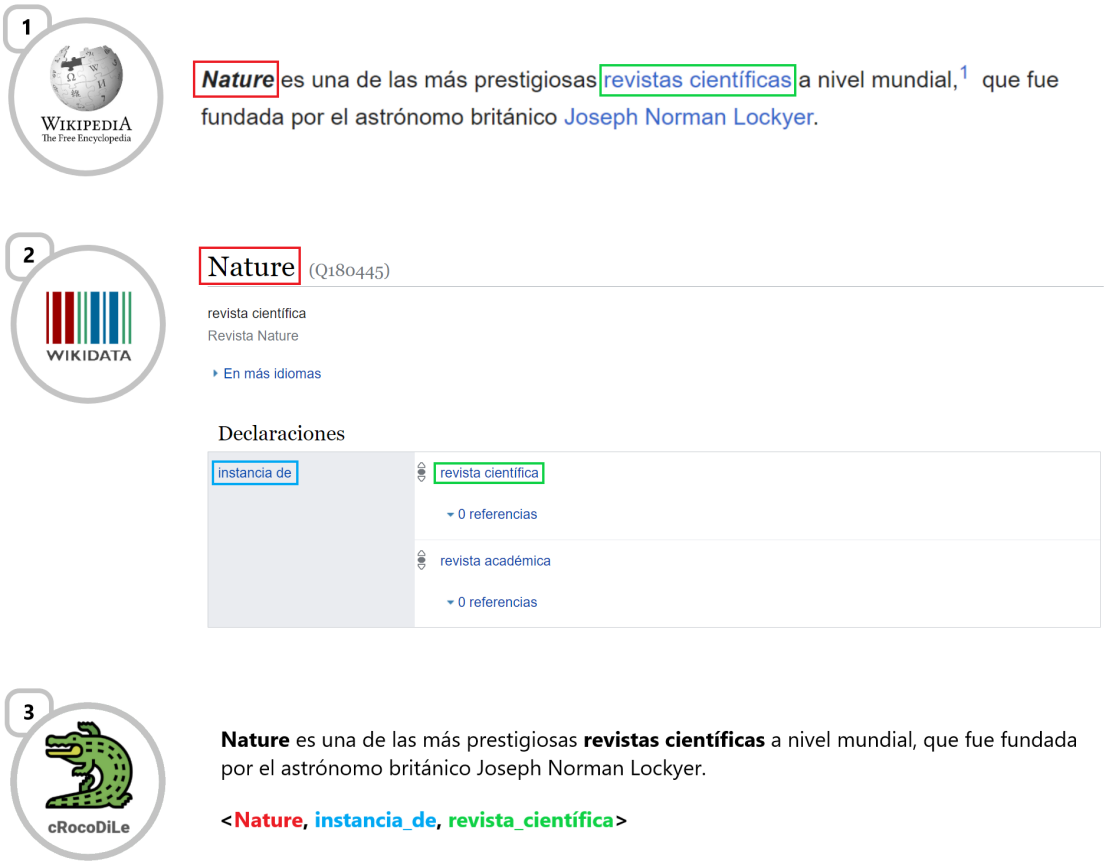

Figura 3.3: Diagrama que ilustra el funcionamiento de cRocoDiLe.

3.2.2. Ejemplo de instancia del corpus

La extensión y formato de nuestro corpus generado es de JSONL (JSON Lines), que sustancialmente es igual que JSON pero permite que cada línea del fichero contenga un objeto JSON independiente, sin estar subordinados a un elemento raíz. Esto permite cargar y leer cada uno de los elementos de forma secuencial y progresiva, en vez de cargar el archivo entero monolíticamente, lo que lo hace especialmente útil cuando se trabaja con grandes cantidades de datos.
Estos elementos son instancias que se usarán para el entrenamiento. Cada instancia tiene 6 pares clave-valor, que son:

- **docid**: identificador único del artículo en Wikipedia
- **title**: título del artículo
- **uri**: identificador único para el artículo en Wikidata
- **text**: texto en crudo, es el resumen del artículo de Wikipedia

- **entities**: vector con las entidades del texto que tienen hipervínculo
- **triples**: vector con los tripletes presentes (sujeto, predicado, objeto)

Ahora, a modo de ejemplo, enseñamos una instancia completa, la misma de la Figura 3.3:


```
{
    "docid": "3328953",
    "title": "Nature",
    "uri": "Q180445",
    "text": "Nature es una de las más prestigiosas revistas científicas
             a nivel mundial, que fue fundada por el astrónomo británico
             Joseph Norman Lockyer.",
    "entities": [
        {
            "uri": "Q5633421",
            "boundaries": [
                38,
                58
            ],
            "surfaceform": "revista científica",
            "annotator": "Me"
        },
        {
            "uri": "Q180445",
            "boundaries": [
                0,
                6
            ],
            "surfaceform": "Nature",
            "annotator": "Me"
        }
    ],
    "triples": [
        {
            "subject": {
                "uri": "Q180445",
                "boundaries": [
                    0,
                    6
                ],
                "surfaceform": "Nature",
                "annotator": "Me"
            },
            "predicate": {
                "uri": "P31",
                "boundaries": null,
                "surfaceform": "instancia de",
                "annotator": "NoSubject-Triple-aligner"
            },
            "object": {
                "uri": "Q5633421",
                "boundaries": [
                    38,
                    58
                ],
```

```
                "surfaceform": "revista científica",
                "annotator": "Me"
            },
            "sentence_id": 1,
            "dependency_path": null,
            "confidence": null,
            "annotator": "NoSubject-Triple-aligner"
        }
    ]
}
```


3.2.3. Cambios necesarios en la librería para la tarea

Wikimapper descarga ficheros de volcado pero no todos los que necesitamos, hay que modificarlo para que incluya un fichero XML comprimido que de otra forma habría que descargar a mano. Esto implica que no podemos utilizar el Wikimapper estándar de PyPi (instalado a través de *pip*) sino que tenemos que clonar el repositorio original para hacerle las oportunas modificaciones.

A su vez, esto cambia la forma en la que se invoca al programa, puesto que ya no se puede llamar modularmente. Hay que modificar la sintaxis de estas llamadas en *extract_lan.sh*.

Los pasos técnicos concretos se detallan en el repositorio público de GitHub que hemos creado para este TFG. Desde ahí se pueden seguir los pasos para replicar nuestros resultados.
Se puede acceder desde `https://github.com/gonzalo-rosae/TFG`.

3.3. REBEL

REBEL [1], acrónimo de «Relation Extraction By End-to-end Language generation» es un modelo para la extracción de relaciones con una arquitectura seq2seq autorregresiva basado en BART [20].

BART (*Bidirectional and AutoRegressive Transformers*) es un modelo de lenguaje basado en la arquitectura Transformer, que fue propuesto por Facebook AI en 2020. La particularidad de BART con respecto a un modelo Transformer se basa en su preentrenamiento. BART ha sido preentrenado en una tarea de eliminación de ruido (*denoising*). Durante el proceso de preentrenamiento, se corrompen los datos de entrada al introducir ruido aleatorio o enmascarar ciertos *tokens*, y luego se entrena al modelo para que reconstruya la entrada original. Este ruido se consigue haciendo transformaciones a la entrada, que incluyen enmascaramiento y eliminación de *tokens*, permutación de frases, rotación de documento o relleno de texto; como se ve en la Figura 3.4.

A continuación se explican los fundamentos de la arquitectura Transformer que se aprecian en la Figura 3.5.

BART utiliza un enfoque de codificador-decodificador clásico, donde el codificador lee y procesa la entrada y el decodificador genera la salida. La arquitectura de BART se compone de varias capas de bloques de atención multicabeza y capas de alimentación hacia adelante.

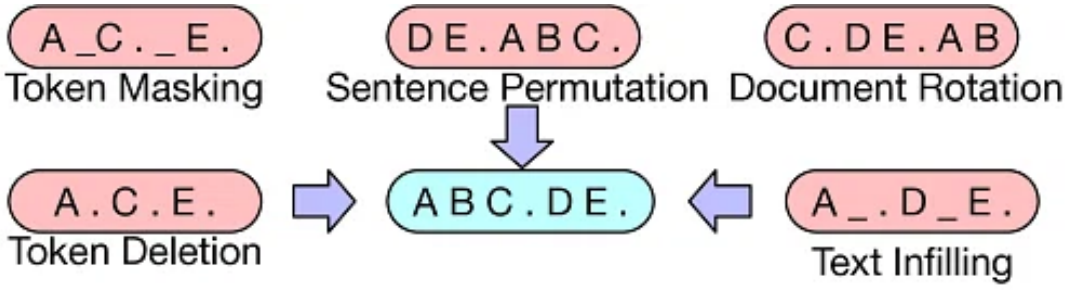

Figura 3.4: Técnicas de preentrenamiento usadas en BART

El codificador toma una secuencia de *tokens* de entrada y la procesa en múltiples capas de atención multicabeza. Cada capa de atención multicabeza calcula diferentes representaciones contextuales de las palabras en función de las relaciones entre los *tokens*. Las capas de alimentación hacia adelante, que son capas completamente conectadas, también se utilizan para procesar la información dentro del codificador.

El decodificador, por otro lado, genera la salida secuencialmente utilizando una estructura autorregresiva. En cada paso de decodificación, se predice el siguiente *token* en función de los *tokens* previamente generados y la representación contextual de la entrada. Esto se hace a través de la atención sobre el codificador y la atención autorregresiva dentro del decodificador. El proceso de generación continúa hasta que se alcanza un *token* de finalización o se alcanza una longitud máxima predefinida.

En definitiva, lo que hace diferente a REBEL y explica su rendimiento final son tres aspectos fundamentales:

1. **Naturaleza BART:** dado que REBEL se basa en BART, hereda todas sus características que le diferencian de otros modelos basados en Transformers. Estas características son arquitectura del decodificador bidireccional, preentrenamiento dual (autorregresivo y de reconstrucción) y uso de codificación de oraciones, que, superando los tokens, lo permiten trabajar a nivel de oraciones o pares de oraciones
2. **Descomposición de tripletes:** REBEL adopta una descomposición simple de tripletes en una secuencia de texto. Esto significa que el texto de entrada se descompone en tripletes, donde cada triplete consta de un sujeto, una relación y un objeto. Esta descomposición simplifica el proceso de extracción y permite un entrenamiento y ajuste fino eficientes.
3. **Conjunto de datos supervisados a distancia a gran escala:** gran parte de su rendimiento se debe a su material de entrenamiento, que es un *dataset* supervisado a distancia a gran escala. Este *dataset* se obtiene aprovechando un modelo de inferencia de lenguaje natural. La supervisión a distancia permite que el modelo aprenda a partir de una gran cantidad de datos sin requerir etiquetas anotadas manualmente.

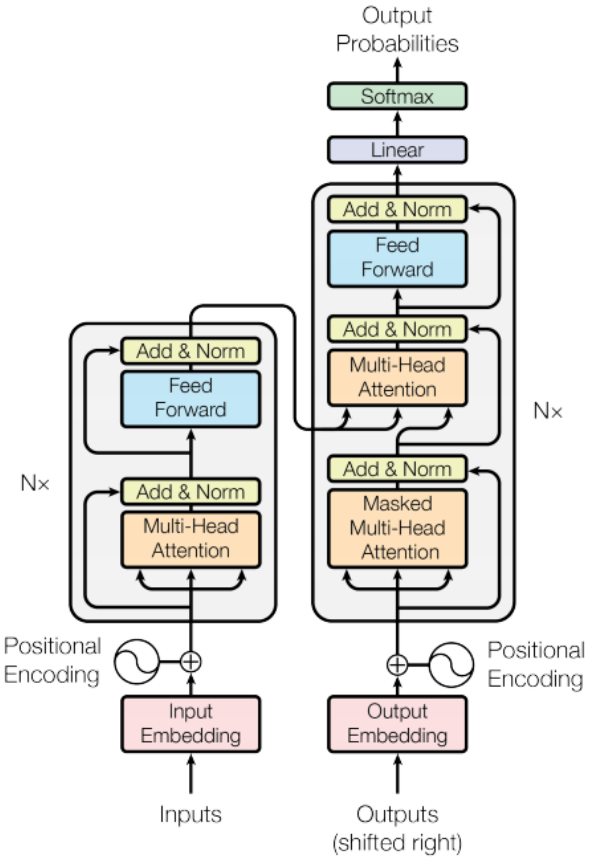

Figura 3.5: Arquitectura Transformer

3.3.1. Instalación, entorno y paquetes necesarios

Todos los problemas relativos a la configuración del entorno vienen dados por la instalación de las diferentes librerías y tecnologías que se utilizan para el funcionamiento de REBEL. Debido a que estas librerías están en boga, no paran de ser actualizadas, y esto incrementa las posibilidades de conflictos entre ellas. Es por eso que, sin ir más lejos, se han tenido que hacer ciertas modificaciones a la configuración que ofrece REBEL por defecto para evitar estos problemas. En la siguiente subsección tratamos esto de manera concreta.

A continuación, esbozaremos los pasos a seguir para, partiendo de cero, poder entrenar un modelo REBEL.

Los autores de REBEL han hecho público un repositorio en GitHub con el modelo y entorno en el que se desplegará[5]. GitHub es una plataforma *online* para alojar y gestionar proyectos de desarrollo de software utilizando Git. A su vez, Git es un sistema de control de versiones distribuido ampliamente utilizado en el desarrollo de software. Para instalar Git se puede seguir la documentación oficial[6]. En GitHub se puede crear una cuenta gratuita, pero para clonar el repositorio de REBEL no es necesario.

Desde ahí se puede clonar el repositorio para trabajar con él de forma local. Dentro

[5]https://github.com/Babelscape/rebel
[6]https://github.com/git-guides/install-git

del mismo, los *scripts* están escritos en lenguaje Python, a excepción de uno que prepara el entorno de ejecución y que está escrito en Bash.

Lo primero que se ha de saber es que el modelo ha sido desarrollado y pensado para ser ejecutado en un entorno Linux. Esto se pone de evidencia al ver cómo todas las rutas están en dicho formato o por la presencia de un fichero Bash (*setup.sh*) , que además es el encargado de configurar el entorno. A pesar de todo, también se puede adaptar a otros sistemas como Windows.

Tal y como indican en el *README.md*, para configurar el intérprete Python se utiliza Conda. Conda es una herramienta de gestión de paquetes y entornos de desarrollo, que permite crear y administrar entornos virtuales con diferentes configuraciones de paquetes y dependencias.

Por tanto, para hacer una réplica estándar se ha de tener instalado este programa en la máquina usuaria. Para su instalación en cualquier sistema operativo se puede seguir la documentación oficial[7]. También se puede optar por no utilizar Conda, en cuyo caso, simplemente habría que ignorar o bien sustituir los comandos Conda por sus equivalentes para el sistema necesario.

En la raíz del repositorio, ejecutando el susodicho *setup.sh* se crea un entorno Conda y se instala *pytorch* junto a las dependencias de *requirements.txt*. Para el entrenamiento del modelo es necesario instalar una variedad de librerías Python, aquellas que vienen especificadas en el fichero *requirements.txt* del directorio raíz. Antes de entrenar al modelo y después de haber preparado el entorno y descargado y colocado el/los *dataset(s)* donde se haya considerado, hay que modificar las rutas oportunas en los archivos de configuración. Estos archivos son de extensión *.yaml* y se encuentran en la carpeta «conf» de la raíz del repositorio.

3.3.2. Cambios necesarios en el repositorio para la tarea

Para empezar, uno de los problemas que hemos tenido es con las versiones de las librerías Python. Si se instalan y descargan tal y como se indica en el fichero *requirements.txt*, surgen problemas de incompatibilidad entre las librerías de «transformers» y «datasets». Esta incompatibilidad puede resolverse disminuyendo la versión de «transformers» o aumentando la de «datasets». Después de investigar y probar diferentes posibilidades, hemos encontrado que la mejor solución es bajar «transformers» de la versión originalmente especificada, 4.19.2, a la 4.5.1.

Por otro lado, hemos tenido que incluir la previamente ausente librería «packaging», en la versión 21.3 para las correctas instalación y ejecución.

De nuevo, para acceder a los detalles técnicos y pasos para la replicación, referimos al mismo enlace. Acceso en `https://github.com/gonzalo-rosae/TFG`.

3.4. Entrenamiento del modelo

Para ejecutar el entrenamiento final del modelo, debido a que estamos trabajando con una gran cantidad de datos y el proceso es computacionalmente costoso, hemos

[7] https://docs.Conda.io/projects/Conda/en/latest/user-guide/install/index.html

usado Magerit. Magerit (concretamente Magerit-3) es un sistema de cómputo científico destinado a la ejecución de cargas de trabajo científico que requieren cómputo intensivo. Magerit pertenece a la Universidad Politécnica de Madrid, que nos ha permitido el acceso para este particular. Este superordenador se alberga en el CeSViMa (**Ce**ntro de **S**upercomputación y **Vi**sualización de **M**adrid), situado en Parque Científico y Tecnológico de la UPM, en Madrid.

Magerit-3 es un clúster compuesto por 72 nodos Lenovo ThinkSystem SD530, cada uno de los cuales dispone de 2 procesadores Intel® Xeon® Gold 6230 de 20 cores @ 2.10 GHz (1.344 GFLOPS) y 192 GiB de RAM, y 48 nodos Lenovo ThinkSystem SD530 (cada uno de los cuales dispone de 2 procesadores Intel® Xeon® Gold 6240R de 24 cores @ 2.40 GHz (1.344 GFLOPS) y 768 GiB de RAM). En la página web oficial[8] hay más información al respecto.

Esto nos ha permitido agilizar mucho el proceso, que, de otra forma habría sido mucho más lento y pesado. La única desventaja es que al ejecutarse en otro entorno hay que adaptar todo a este. Se ha tenido que dedicar cierto tiempo a entender cómo interactuar con el mismo. Para esto, existe una documentación oficial[9] que se ha usado como referencia.

La interacción es a través de la consola de comandos, utilizando el protocolo SSH (*Secure SHell*). Evidentemente para conectarse hay que utilizar un usuario y contraseña que han sido previamente dados de alta en el sistema con una serie de permisos.

Para simplificar el uso de las aplicaciones instaladas en Magerit se utiliza Lmod, una implementación de *Environment Modules*. Esta utilidad se encarga de preparar el entorno de ejecución para utilizar distintas versiones de las aplicaciones y sus dependencias mediante la carga de módulos de configuración. O sea, que estas simplemente se cargan y descargan, entendido como activación y desactivación. Magerit se explota mediante trabajos *batch* usando SLURM como gestor y planificador de recursos. Para ejecutar se indican las características del trabajo que se necesita y el sistema se encargará de reservar los recursos necesarios y ejecutar las tareas. Estos parámetros se pueden modificar mediante las llamadas directivas de SLURM.

Tanto las directivas como el código propiamente dicho se especifican en un archivo de procesamiento por lotes, concretamente ficheros Bash (*.sh*).

[8] https://www.cesvima.upm.es/infrastructure/magerit
[9] https://docs.cesvima.upm.es/magerit/

Capítulo 4

Resultados y conclusiones

En este capítulo abarcamos los resultados obtenidos y hacemos un análisis que nos conduce a las conclusiones y que proyectan el camino a seguir para seguir trabajando en este campo.

4.1. Resultados

Ausencia de corpus de ER en español que sean de calidad: después de haber hecho una búsqueda durante la etapa inicial del trabajo, hemos concluido satisfactoriamente que a fecha de realización del trabajo, no existen *datasets* en español para ER que cumplan los requisitos ya desarrollados en 3.1.2.

Creación del *dataset* en español usando cRocoDiLe: siguiendo la metodología de cRocoDiLe y haciendo uso de esta herramienta, hemos marcado los pasos para crear el *dataset* en español. El aporte y resultado más importante que hemos conseguido ha sido el de resolver numerosos problemas que en la práctica aparecían al intentar usar cRocoDiLe tal y como se describía en su repositorio. Como ya se ha explicado, hemos creado además un repositorio público en GitHub en el que hemos detallado paso a paso los comandos a ejecutar para hacer los ajustes oportunos a cRocoDiLe y preparar el entorno para su ejecución. De esta forma, aspiramos a ayudar al máximo número de investigadores y desarrolladores que quieran replicarlo.

Entrenamiento de prueba con *dataset* en inglés: a la par de empezar a trabajar con cRocoDiLe y en la corrección de sus problemas, hicimos también algunas pruebas con el modelo de REBEL. Estas consistieron en intentar replicar los pasos y resultados que obtuvieron los investigadores en el paper en el que lo presentaban. Para esto, utilizamos uno de los *datasets* de referencia para la ER: CONLL04.

4.1.1. Creación del *dataset* con cRocoDiLe

En el marco de esta investigación, se ha llevado a cabo la creación de un *dataset* específico para el español utilizando la herramienta cRocoDiLe, siguiendo sus especificaciones. El objetivo principal ha sido construir un corpus anotado con relaciones semánticas para el idioma español. Aunque el *dataset* aún no ha sido completado en su totalidad debido al tiempo limitado disponible, se han obtenido resultados significativos hasta el momento.

El *dataset* en desarrollo abarca un amplio rango de textos en español, y se espera que contenga diversas relaciones semánticas. Hasta la fecha de entrega de esta memoria, se ha logrado recolectar una cantidad considerable de datos, aunque no se dispone de cifras exactas debido a la fase de desarrollo activa del proyecto. En la defensa se exhibirán ejemplos representativos del *dataset* y se detallarán las relaciones semánticas que han sido identificadas y anotadas hasta el momento.

Una vez preparado todo el entorno, con los cambios indicados oportunos, al ejecutar el *script extract_lan.sh* seguido del código de idioma «es» (español) en la raíz del repositorio cRocoDiLe, se creará una carpeta de ruta «data/es/». Aquí comenzarán a descargarse una serie de ficheros que interactuarán entre sí para generar el corpus final. Entre ellos, está el fichero más pesado, aquel que contiene todos los artículos de la Wikipedia en español. Este está comprimido en formato bzip2 (extensión *.bz2*), que envuelve un fichero XML (*eXtensible Markup Language*), sucedáneo de JSON, para el almacén de datos. Gracias a estar comprimido, solo ocupa algo más de 4 GB. A modo de comparación, el archivo equivalente de la Wikipedia en inglés (que se puede descargar cambiando el código de idioma a «en») ocupa poco más de 20 GB. De este archivo es donde se extraen los resúmenes con Wikiextractor, que se enlazarán con Wikidata, etc.

Al terminar la ejecución, también se habrá creado la carpeta hermana de ruta «out/es/», donde se almacenan organizados por carpetas en orden alfabético los ficheros de entrenamiento. Estos están codificados en formato JSONL. Si también se ha ejecutado el paso de filtrado de los tripletes con el modelo de NLI, estos archivos depurados se guardarán en la carpeta homóloga «out_clean/es». Estos últimos JSONL serán los que finalmente consumirá nuestro modelo cuando le hagamos el ajuste fino.

4.1.2. Entrenamiento del modelo con *dataset* en inglés

Para probar el entrenamiento del modelo propuesto, se seleccionó uno de los *datasets* de ER de referencia en inglés que se explicaron en la sección 3.1. Concretamente utilizamos CONLL04 (*Conference on Computational Natural Language Learning*). Este *dataset* se utilizó como referencia para probar el funcionamiento del modelo.

Durante el proceso de entrenamiento, se registraron los resultados en un archivo de registro o *log* y se obtuvieron los resultados esperados. Los valores para las diferentes métricas eran similares a los especificados en el *paper* de REBEL [1] para este *dataset*. Se puede apreciar la comparación en la Tabla 4.1.

Estos resultados demostraron el éxito del entrenamiento del modelo. Se realizaron diversas evaluaciones utilizando métricas estándar, como *precision*, *recall*, *F1-score* y *accuracy*, para evaluar el rendimiento del modelo en la tarea de extracción de relaciones semánticas. Estas métricas son ampliamente utilizadas en la evaluación de modelos de lenguaje y proporcionan una medida cuantitativa de su desempeño. A continuación, se describe brevemente en qué consiste cada una de ellas:

- ***Precision***: Mide la proporción de instancias positivas clasificadas correctamente. Es útil para evaluar la precisión de las predicciones positivas.
- ***Recall***: Mide la proporción de instancias positivas que son correctamente identificadas. Es útil para evaluar la capacidad del modelo para encontrar todas las instancias positivas.

	Precision	**Recall**	**F1**
Original	77.53	74.20	76.13
Réplica	78.57	73.11	75.74

Cuadro 4.1: Tabla comparativa de métricas

- ***F1-score***: Es una medida que combina *precision* y *recall* en un solo valor. Proporciona una medida equilibrada del rendimiento del modelo.
- ***Accuracy***: Mide la proporción general de instancias clasificadas correctamente. Es útil para evaluar el rendimiento general del modelo.

La reproducción exitosa de los resultados esperados en este corpus de referencia en inglés, CONLL04, es un indicador sólido de que el modelo propuesto ha sido entrenado de manera efectiva. La obtención de valores similares para las diferentes métricas, tal y como se especifica en el paper de REBEL para este *dataset*, demuestra que el modelo funciona de acuerdo a lo planteado por los autores, alcanzando el objetivo de reproducibilidad.

Esto proporciona una base sólida para tener confianza en la capacidad del modelo para entrenar correctamente en otros idiomas, como el español. Al aplicar el mismo proceso de entrenamiento y evaluación en un *dataset* en español, podremos determinar si el modelo es capaz de generalizar y aprender correctamente las relaciones semánticas en ese idioma. La reproducibilidad en el experimento en inglés nos brinda una garantía de que el modelo funcionará de manera similar en otros idiomas, siempre y cuando se siga el mismo procedimiento.

En la próxima iteración de este proyecto, se entrenará el modelo con el nuevo *dataset* creado en español específicamente para este trabajo. Dado que en el momento de redacción de esta memoria el corpus aún no se ha generado por completo. Los resultados que se obtendrán con el entrenamiento del nuevo corpus no deben de distar demasiado con respecto de los que ha obtenido REBEL para los *datasets* en inglés.

4.2. Conclusiones personales

En conclusión, este proyecto ha revelado dos importantes aspectos que han impactado nuestra percepción sobre la extracción de relaciones semánticas —y en general, el procesamiento del lenguaje natural— para el español.

En primer lugar, se ha destacado la complejidad inherente a este tipo de trabajo. Pensando en REBEL, por ejemplo, aunque se haya publicado un artículo científico y se hayan proporcionado repositorios y herramientas para replicar el trabajo, es crucial comprender que la ejecución exitosa de un proyecto así requiere un esfuerzo considerable. No se trata simplemente de presionar un botón y obtener resultados inmediatos. Se han enfrentado desafíos técnicos y conceptuales a lo largo de todo el proceso, lo cual ha demostrado que la implementación práctica de modelos de lenguaje para la extracción de relaciones semánticas es una tarea exigente y rigurosa.

Por otro lado, es sorprendente constatar la falta de recursos disponibles en el ámbito del PLN, específicamente relacionados con el español, considerando que es el segundo

idioma más hablado a nivel mundial en términos de hablantes nativos[1]. Este hallazgo destaca una brecha significativa en cuanto a la disponibilidad de herramientas y datos para el desarrollo de proyectos en esta área. El hecho de que haya escasez de recursos adecuados resalta la necesidad de impulsar la investigación y el desarrollo en el ámbito del PLN para el español, a fin de aprovechar plenamente el potencial de este idioma, especialmente en el ámbito de la extracción de relaciones semánticas.

4.3. Trabajo futuro

El trabajo futuro por hacer en este campo queda marcado por las limitaciones de este trabajo.

El primer problema es relativo al corpus de entrenamiento, que, a pesar de suponer un gran avance como ya se ha explicado en otros apartados de la memoria, sigue teniendo defectos. El primero es ser un *dataset* plateado, o sea, generado automáticamente. Esto comporta que la calidad de sus anotaciones no es óptima, pues pueden colarse relaciones que realmente no están implicadas por el texto, o al revés, relaciones presentes que no son anotadas. Es por este motivo por el que se debería trabajar en la generación de un *dataset* dorado, que haya sido anotado o directamente supervisado por humanos, reduciendo así los errores al mínimo posible.

En esta línea, recientemente los investigadores de REBEL en colaboración con otros han publicado un nuevo *paper*, llamado «RED^{FM}: a Filtered and Multilingual Relation Extraction Dataset» [34]. En este, de nuevo, motivados por la falta de recursos actualizados, gratuitos, de calidad y que no consideren solo el inglés, han creado nuevos *datasets* para la ER. Por un lado, han lanzado SRED^{FM}, un conjunto de datos anotados automáticamente que abarca 18 idiomas (incluyendo español), 400 tipos de relaciones y 13 tipos de entidades, con un total de más de 40 millones de instancias de tripletes. Por otro, RED^{FM}, un conjunto de datos más pequeño y revisado por humanos para siete lenguas (incluyendo español).

[1] https://es.statista.com/estadisticas/635631/los-idiomas-mas-hablados-en-el-mundo/

Capítulo 5

Impacto del trabajo

En este capítulo abarcamos el impacto que tiene la publicación en el dominio público de este trabajo junto a los recursos desarrollados. Por un lado examinaremos el impacto general y por otro nos focalizaremos en los Objetivos de Desarrollo Sostenible de la Agenda 2030.

5.1. Impacto general

En línea con los tres objetivos de los que se compone este trabajo, se consideran tres impactos generales, así como un cuarto más concreto y quizá algo marginal, pero que puede existir.

El primero, relacionado con el primer objetivo («Identificar conjuntos de datos en español que estén anotados con relaciones»), es hacer constar a otros investigadores de que a fecha de publicación de este documento no hay *datasets* en español para la extracción de relaciones con un tamaño considerable y que no sean de un dominio específico. Esto es ya una contribución en tanto que informa del estado actual de los recursos existentes para esta tarea de ER en español, ahorrando tiempo de los investigadores o desarrolladores que quieran encontrar un *dataset* para esto.

El segundo, relacionado oportunamente con el segundo objetivo («Crear un corpus de extracción de relaciones para el español»), y en nuestra estimación el más importante, es el de dotar de este recurso de acceso público y gratuito. La adaptación de la herramienta cRocoDiLe para poder crear el corpus plateado en español de un tamaño y calidad tales será de gran ayuda para futuros investigadores.
Primero, se creará la visibilidad de esta herramienta como específica para el español, ya que muchas veces el multilingüismo de ciertas tecnologías las esconde de sus potenciales usuarios finales, que utilizan una sola lengua.

Por otro lado, la facilidad con la que se puede utilizar este sistema, ya que ha sido depurado y ajustado lo máximo posible para incrementar su usabilidad. A lo largo del trabajo, hemos encontrado muchos más problemas de los que pensábamos para poder utilizar esta herramienta y generar un *dataset* en español. Esto nos ha creado una concienciación sobre la reproducibilidad y uso de las herramientas que se desarrollan en el campo de la investigación, de forma que hemos procurado su máxima facilidad de uso y la menor caducidad posible. Esta última cualidad ha sido tenida en cuenta en consideración del impacto, porque visto el papel que han tenido los

ficheros de volcado de Wikipedia para el entrenamiento de los Grandes Modelos del Lenguaje, creemos que es de vital importancia poder seguir trabajando de esta manera con ella. De hecho, la publicación media de artículos al día entre el 1 de enero de 2023 y el 1 de mayo del mismo año, es de más de 450 en inglés[1] y más de 250 en español[2].
Esto pone en evidencia que en el futuro, se podrán generar *datasets* todavía más grandes que los actuales y mejorar así la calidad de los modelos entrenados.

El tercero, alineado con el tercer objetivo («Usar modelos de lenguaje preentrenados en español y reajustarlos para la tarea de extracción de relaciones»), tiene un beneficio directo en aquellos interesados en utilizar un modelo de ER para español que tenga un rendimiento de estado de la cuestión. Prevemos que esto tendrá un mayor impacto en los desarrolladores que necesiten implementar soluciones que funcionen de forma inmediata, sin tener tanto interés por resultados más precisos o por intentar mejorar el modelo que nosotros lanzamos.

Por último, también puede haber un potencial destinatario *amateur* al que le interese este campo de investigación y quiera aprender más al respecto. En este sentido, nosotros hemos proporcionado un sistema de fácil uso y, sobre todo, una extensa documentación, explicando los aspectos fundamentales de las herramientas para facilitar la comprensión a cualquier usuario final, con cierta independencia de su bagaje técnico sobre el PLN. La documentación e instrucción de nuestro sistema se encuentra repartida entre el presente documento, de carácter más teórico y explicativo; y las notas de uso de cada una de las herramientas en los repositorios GitHub, más enfocadas a la preparación del entorno y el uso inmediato de estas herramientas.

5.2. Objetivos de Desarrollo Sostenible

En relación al **Objetivo 4** de Educación[3], la Meta 4.3 puede ser relevante: «De aquí a 2030, asegurar el acceso igualitario de todos los hombres y las mujeres a una formación técnica, profesional y superior de calidad, incluida la enseñanza universitaria». Aquí se incluye el acceso a información específica y técnica de calidad en Internet. Por lo tanto, se puede destacar la importancia de proporcionar acceso igualitario a recursos educativos en línea, incluyendo este trabajo, para contribuir a la democratización del conocimiento.

En cuanto al **Objetivo 8** de Trabajo y Crecimiento Económico[4], la Meta 8.2 es pertinente: «Lograr niveles más elevados de productividad económica mediante la diversificación, la modernización tecnológica y la innovación, entre otras cosas centrándose en los sectores con gran valor añadido y un uso intensivo de la mano de obra». En este sentido, este trabajo puede promover la modernización tecnológica e innovación al aplicar nuevas técnicas y metodologías en el procesamiento del lenguaje natural, lo cual contribuye a mejorar la productividad y eficiencia en la creación de modelos lingüísticos.

En relación al **Objetivo 9** de Infraestructura[5], varias metas están relacionadas con

[1] https://stats.wikimedia.org//**en**.wikipedia.org/content/pages-to-date
[2] https://stats.wikimedia.org//**es**.wikipedia.org/content/pages-to-date
[3] https://www.un.org/sustainabledevelopment/es/education/
[4] https://www.un.org/sustainabledevelopment/es/economic-growth/
[5] https://www.un.org/sustainabledevelopment/es/education/

el trabajo:

La Meta 9.5: «Aumentar la investigación científica y mejorar la capacidad tecnológica de los sectores industriales de todos los países, en particular los países en desarrollo, entre otras cosas fomentando la innovación y aumentando considerablemente, de aquí a 2030, el número de personas que trabajan en investigación y desarrollo por millón de habitantes y los gastos de los sectores público y privado en investigación y desarrollo». Nuestra investigación y desarrollo en procesamiento del lenguaje natural contribuye a mejorar la capacidad tecnológica de algunos países en desarrollo, especialmente los de habla hispana por el enfoque de nuestro trabajo, fomentando la innovación en el campo de la inteligencia artificial y la lingüística computacional.

La Meta 9.a: «Facilitar el desarrollo de infraestructuras sostenibles y resilientes en los países en desarrollo mediante un mayor apoyo financiero, tecnológico y técnico a los países africanos, los países menos adelantados, los países en desarrollo sin litoral y los pequeños Estados insulares en desarrollo». Nuestro trabajo puede ser aplicado en contextos de países en desarrollo, concretamente en el país africano de Guinea Ecuatorial, en donde el español es lengua oficial y se usa como lengua vehicular. Además, con nuestra resolución de los problemas de cRocoDiLe, hemos facilitado el trabajo de generación de *datasets* en otros idiomas minoritarios. Contribuirá a facilitar el desarrollo de infraestructuras sostenibles en dichos países al proporcionar recursos y tecnologías para el procesamiento y análisis de sus lenguas nativas.

La Meta 9.c: «Aumentar significativamente el acceso a la tecnología de la información y las comunicaciones y esforzarse por proporcionar acceso universal y asequible a Internet en los países menos adelantados de aquí a 2020». Este trabajo contribuye indirectamente a este objetivo al permitir un acceso más amplio a tecnologías de procesamiento del lenguaje en general y a recursos en línea, promoviendo así la inclusión digital y el acceso a la información para comunidades con recursos limitados.

Además de los objetivos y metas específicos mencionados anteriormente, este trabajo presenta aspectos relevantes que se alinean con los anteriores y que vale la pena destacar:

- **Contribución a la diversidad lingüística y desarrollo en países en desarrollo:** Este trabajo puede ser una referencia valiosa y un ejemplo concreto para la creación de recursos en lenguas minoritarias o en países en desarrollo que carecen de conjuntos de datos anotados adecuados como ya se ha mencionado. Al abordar estas necesidades, se fomenta la inclusión y se promueve el acceso a la tecnología del procesamiento del lenguaje natural en comunidades que a menudo se encuentran marginadas en este ámbito.
- **Promoción de la ciencia abierta y la reproducibilidad:** La investigación llevada a cabo en este trabajo se adhiere a los principios de la ciencia abierta. Se proporciona una descripción detallada de los métodos y técnicas utilizados, lo que permite a otros investigadores reproducir y verificar los resultados obtenidos. Al compartir el conocimiento y fomentar la colaboración, se impulsa el avance de la investigación en el campo del procesamiento del lenguaje natural.
- **Consideraciones ambientales y eficiencia energética:** Un aspecto relevante de este trabajo es su enfoque en la eficiencia energética y la reducción del impacto ambiental al haber utilizado un modelo preentrenado como BART. El hecho de hacer un ajuste fino sobre un modelo preentrenado supone un ahorro

energético enorme con respecto de la creación de uno nuevo de cero. De esta forma estamos aprovechando el impacto ambiental que ya irremediablemente tuvo el modelo en su entrenamiento. Esto adquiere especial importancia en países en desarrollo, donde los recursos energéticos son limitados y los problemas ambientales requieren una atención prioritaria. Al buscar soluciones más eficientes y sostenibles, este trabajo contribuye a un desarrollo tecnológico responsable.

- **Abordaje de sesgos de género en los conjuntos de datos:** Al utilizar grafos y bases de datos como Wikidata, se establece una estrategia para mitigar los sesgos de género presentes en los conjuntos de datos utilizados en la investigación. Al garantizar una representación equitativa y no sesgada de hombres y mujeres, se promueve la igualdad de género y se evita la reproducción de estereotipos perjudiciales.

Estos aspectos subrayan la importancia y el impacto potencial de este trabajo en términos de diversidad lingüística, ciencia abierta, consideraciones ambientales e igualdad de género en el contexto del PLN.

Bibliografía

[1] Pere-Lluıs Huguet Cabot y Roberto Navigli. «REBEL: Relation Extraction By End-to-end Language generation». En: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Punta Cana, Dominican Republic: Association for Computational Linguistics, nov. de 2021, págs. 2370-2381. DOI: 10.18653/v1/2021.findings-emnlp.204. URL: https://aclanthology.org/2021.findings-emnlp.204.

[2] Bruno Taillé y col. «Let's Stop Incorrect Comparisons in End-to-end Relation Extraction!» En: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, nov. de 2020, págs. 3689-3701. DOI: 10.18653/v1/2020.emnlp-main.301. URL: https://aclanthology.org/2020.emnlp-main.301.

[3] William Hogan. *An Overview of Distant Supervision for Relation Extraction with a Focus on Denoising and Pre-training Methods*. 2022. arXiv: 2207.08286 [cs.CL].

[4] Tomas Mikolov y col. *Distributed Representations of Words and Phrases and their Compositionality*. 2013. arXiv: 1310.4546 [cs.CL].

[5] Jeffrey Pennington, Richard Socher y Christopher Manning. «GloVe: Global Vectors for Word Representation». En: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, oct. de 2014, págs. 1532-1543. DOI: 10.3115/v1/D14-1162. URL: https://aclanthology.org/D14-1162.

[6] Daojian Zeng y col. «Relation Classification via Convolutional Deep Neural Network». En: *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Dublin, Ireland: Dublin City University y Association for Computational Linguistics, ago. de 2014, págs. 2335-2344. URL: https://aclanthology.org/C14-1220.

[7] Yuhao Zhang y col. «Position-aware Attention and Supervised Data Improve Slot Filling». En: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, sep. de 2017, págs. 35-45. DOI: 10.18653/v1/D17-1004. URL: https://aclanthology.org/D17-1004.

[8] Makoto Miwa y Yutaka Sasaki. «Modeling Joint Entity and Relation Extraction with Table Representation». En: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, oct. de 2014, págs. 1858-1869. DOI: 10.3115/v1/D14-1200. URL: https://aclanthology.org/D14-1200.

[9] Sachin Pawar, Pushpak Bhattacharyya y Girish Palshikar. «End-to-end Relation Extraction using Neural Networks and Markov Logic Networks». En: *Proceedings of the 15th Conference of the European Chapter of the Association for*

Computational Linguistics: Volume 1, Long Papers. Valencia, Spain: Association for Computational Linguistics, abr. de 2017, págs. 818-827. URL: https://aclanthology.org/E17-1077.

[10] Jue Wang y Wei Lu. «Two are Better than One: Joint Entity and Relation Extraction with Table-Sequence Encoders». En: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, nov. de 2020, págs. 1706-1721. DOI: 10.18653/v1/2020.emnlp-main.133. URL: https://aclanthology.org/2020.emnlp-main.133.

[11] Arzoo Katiyar y Claire Cardie. «Going out on a limb: Joint Extraction of Entity Mentions and Relations without Dependency Trees». En: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, jul. de 2017, págs. 917-928. DOI: 10.18653/v1/P17-1085. URL: https://aclanthology.org/P17-1085.

[12] Heike Adel e Hinrich Schütze. «Global Normalization of Convolutional Neural Networks for Joint Entity and Relation Classification». En: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, sep. de 2017, págs. 1723-1729. DOI: 10.18653/v1/D17-1181. URL: https://aclanthology.org/D17-1181.

[13] Markus Eberts y Adrian Ulges. «Span-based Joint Entity and Relation Extraction with Transformer Pre-training». En: *ArXiv* abs/1909.07755 (2019).

[14] Zhenzhong Lan y col. «ALBERT: A Lite BERT for Self-supervised Learning of Language Representations». En: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=H1eA7AEtvS.

[15] Markus Eberts y Adrian Ulges. «An End-to-end Model for Entity-level Relation Extraction using Multi-instance Learning». En: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics, abr. de 2021, págs. 3650-3660. DOI: 10.18653/v1/2021.eacl-main.319. URL: https://aclanthology.org/2021.eacl-main.319.

[16] Xiangrong Zeng y col. «Extracting Relational Facts by an End-to-End Neural Model with Copy Mechanism». En: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, jul. de 2018, págs. 506-514. DOI: 10.18653/v1/P18-1047. URL: https://aclanthology.org/P18-1047.

[17] Daojian Zeng, Haoran Zhang y Qianying Liu. «CopyMTL: Copy Mechanism for Joint Extraction of Entities and Relations with Multi-Task Learning». En: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.05 (abr. de 2020), págs. 9507-9514. DOI: 10.1609/aaai.v34i05.6495. URL: https://ojs.aaai.org/index.php/AAAI/article/view/6495.

[18] Tapas Nayak y Hwee Tou Ng. «Effective Modeling of Encoder-Decoder Architecture for Joint Entity and Relation Extraction». En: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.05 (abr. de 2020), págs. 8528-8535. DOI: 10.1609/aaai.v34i05.6374. URL: https://ojs.aaai.org/index.php/AAAI/article/view/6374.

[19] Xiangrong Zeng y col. «Learning the Extraction Order of Multiple Relational Facts in a Sentence with Reinforcement Learning». En: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, nov. de 2019, págs. 367-377. DOI: 10.18653/v1/D19-1035. URL: https://aclanthology.org/D19-1035.

[20] Mike Lewis y col. «BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension». En: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, jul. de 2020, págs. 7871-7880. DOI: 10.18653/v1/2020.acl-main.703. URL: https://aclanthology.org/2020.acl-main.703.

[21] Colin Raffel y col. «Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer». En: *Journal of Machine Learning Research* 21.140 (2020), págs. 1-67. URL: http://jmlr.org/papers/v21/20-074.html.

[22] Nicola De Cao y col. «Autoregressive Entity Retrieval». En: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=5k8F6UU39V.

[23] Michele Bevilacqua, Rexhina Blloshmi y Roberto Navigli. «One SPRING to Rule Them Both: Symmetric AMR Semantic Parsing and Generation without a Complex Pipeline». En: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.14 (mayo de 2021), págs. 12564-12573. DOI: 10.1609/aaai.v35i14.17489. URL: https://ojs.aaai.org/index.php/AAAI/article/view/17489.

[24] Rexhina Blloshmi y col. «Generating Senses and RoLes: An End-to-End Model for Dependency- and Span-based Semantic Role Labeling». En: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. Ed. por Zhi-Hua Zhou. Main Track. International Joint Conferences on Artificial Intelligence Organization, ago. de 2021, págs. 3786-3793. DOI: 10.24963/ijcai.2021/521. URL: https://doi.org/10.24963/ijcai.2021/521.

[25] Michele Bevilacqua, Marco Maru y Roberto Navigli. «Generationary or "How We Went beyond Word Sense Inventories and Learned to Gloss"». En: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, nov. de 2020, págs. 7207-7221. DOI: 10.18653/v1/2020.emnlp-main.585. URL: https://aclanthology.org/2020.emnlp-main.585.

[26] Mariona Taulé, M. Antònia Martı y Marta Recasens. «AnCora: Multilevel Annotated Corpora for Catalan and Spanish». En: *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*. Marrakech, Morocco: European Language Resources Association (ELRA), mayo de 2008. URL: http://www.lrec-conf.org/proceedings/lrec2008/pdf/35_paper.pdf.

[27] Alisa Zhila y Alexander Gelbukh. «Open Information Extraction for Spanish Language based on Syntactic Constraints». En: *Proceedings of the ACL 2014 Student Research Workshop*. Baltimore, Maryland, USA: Association for Computational Linguistics, jun. de 2014, págs. 78-85. DOI: 10.3115/v1/P14-3011. URL: https://aclanthology.org/P14-3011.

[28] Hermenegildo Fabregat. «Biomedical Information Extraction: Exploring new entities and relationships». En: 2021. URL: https://dialnet.unirioja.es/servlet/tesis?codigo=292117.

[29] Alessandro Seganti y col. «Multilingual Entity and Relation Extraction Dataset and Model». En: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics, abr. de 2021, págs. 1946-1955. DOI: 10.18653/v1/2021.eacl-main.166. URL: https://aclanthology.org/2021.eacl-main.166.

[30] Dan Roth y Wen-tau Yih. «A Linear Programming Formulation for Global Inference in Natural Language Tasks». En: *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*. Boston, Massachusetts, USA: Association for Computational Linguistics, mayo de 2004, págs. 1-8. URL: https://aclanthology.org/W04-2401.

[31] Yuan Yao y col. «DocRED: A Large-Scale Document-Level Relation Extraction Dataset». En: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, jul. de 2019, págs. 764-777. DOI: 10.18653/v1/P19-1074. URL: https://aclanthology.org/P19-1074.

[32] Sebastian Riedel, Limin Yao y Andrew McCallum. «Modeling Relations and Their Mentions without Labeled Text». En: *Machine Learning and Knowledge Discovery in Databases*. Ed. por José Luis Balcázar y col. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, págs. 148-163. ISBN: 978-3-642-15939-8.

[33] Mike Mintz y col. «Distant supervision for relation extraction without labeled data». En: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Suntec, Singapore: Association for Computational Linguistics, ago. de 2009, págs. 1003-1011. URL: https://aclanthology.org/P09-1113.

[34] Pere-Lluís Huguet Cabot y col. *RED$^{\text{FM}}$: a Filtered and Multilingual Relation Extraction Dataset*. 2023. arXiv: 2306.09802 [cs.CL].