Detección de sujetos omitidos en español aplicando técnicas de traducción automática

Informe de Proyecto de Grado presentado por

José Diego Suárez Hernández

en cumplimiento parcial de los requerimientos para la graduación de la carrera de Ingeniería en Computación de Facultad de Ingeniería de la Universidad de la República

Supervisor

Luis Chiruzzo

Montevideo, 2 de agosto de 2023

Agradecimientos

A Luis, supervisor de este trabajo, cuyos consejos y aliento fueron fundamentales para la conclusión de este proyecto, y a mis padres, Andrea y José, por su apoyo infinito.

Resumen

Este estudio aborda el problema de la detección de cláusulas con sujetos omitidos y verbos impersonales en español adaptando una metodología de traducción automática. Se utilizó el framework de traducción automática neuronal OpenNMT, basado en redes neuronales recurrentes LSTM con mecanismos de atención, para construir un modelo de traducción secuencia-a-secuencia, el cual fue entrenado sobre un corpus paralelo derivado de la versión en español del corpus anotado AnCora consistente en oraciones en texto plano emparejadas con secuencias de identificadores de la clase gramatical (POS) de cada lexema y etiquetas especiales para marcar la presencia de tres fenómenos distintos: la ocurrencia de verbos con sujeto explícito, verbos con sujeto implícito u omitido y verbos en un uso impersonal. A partir de estos modelos se elaboró clasificadores orientados a detectar la ocurrencia de estos fenómenos para cada verbo finito en la entrada, a través de la traducción al formato de salida y la extracción de las etiquetas relevantes.

Durante el desarrollo de este trabajo se investigó la efectividad de múltiples formatos de salida y configuraciones hiperparamétricas para los modelos de traducción automática y se evaluó el rendimiento de los clasificadores resultantes tanto respecto a su capacidad para identificar correctamente instancias con sujetos omitidos, explícitos y verbos impersonales como respecto al tiempo de cómputo requerido para el entrenamiento y aplicación de los modelos propuestos.

Los resultados obtenidos demostraron la viabilidad de este enfoque, obteniéndose resultados cercanos a los del estado del arte para la detección de sujetos omitidos y, en particular, una medida F1 de 0,7685 para el reconocimiento de sujeto omitido en un escenario de clasificación ternaria (frente a sujeto explícito y verbos en uso impersonal) que supera a los valores obtenidos en los antecedentes publicados. Se constató asimismo que los requisitos de cómputo para el entrenamiento de los modelos fueron moderados, obteniéndose los rendimientos más altos para un modelos con redes de dos capas ocultas que solamente requirieron dos horas de entrenamiento en una GPU de *entry-level* por lo que se identificó además el potencial de la adaptación de mecanismos de traducción automática como una alternativa computacionalmente eficiente para el reconocimiento de características en problemas de análisis lingüístico.

Palabras clave: Sujeto omitido, PLN, Traducción automática, OpenNMT

Índice general

Capítulo 1

Introducción

Las oraciones en el idioma español en general se componen de dos elementos sintácticos, sujeto y predicado, donde el sujeto es un argumento que se describe o participa en una acción definida por el predicado. A modo de ejemplo, considérese la oración 1:

- (1) ***La niña*** *lee un libro.*

En esta oración, se identifica a *"la niña"* como sujeto, sobre el cual versa el predicado *"lee un libro"*. Este mismo análisis aplica a otras lenguas como el inglés y el francés, donde en la estructura oracional también se identifica un sujeto (en negrita) y un predicado:

- (1-inglés) ***The girl*** *is reading a book.*
- (1-francés) ***La fille*** *lit un livre.*

En el español, resulta posible omitir el sujeto de una oración. En la oración 2 se identifica únicamente un predicado, el cual versa sobre un sujeto que el interlocutor podrá eventualmente inferir en base al contexto de la conversación o la concordancia de persona y número en la conjugación verbal.

- (2) *Lee un libro.*

Este fenómeno del español, conocido como sujeto omitido, sujeto tácito o *pro-drop* (del inglés *pronoun dropping*, omisión de pronombre), se presenta en otras lenguas como el portugués, el ruso y el mandarín pero no se permite en idiomas como el inglés y el francés donde el sujeto debe explicitarse al menos mediante la inclusión de un pronombre, debiéndose indicar, por ejemplo, que *"**ella**"* (la niña) es quien *"lee un libro"*:

- (2-inglés) ***She*** *is reading a book.*
- (2-francés) ***Elle*** *lit un livre.*

En el español se presentan además casos donde no solo no se encuentra un sujeto explícito sino que no es posible siquiera proponer un sujeto significativo para el predicado, como ocurre en las oraciones 3 y 4:

- (3) *Se descansa bien en ese hotel.*
- (4) *Llovía muy fuerte.*

En los análisis gramaticales del español (Real Academia Española, 2011), se considera que los verbos "descansa" y "llovía" en estas oraciones son impersonales y que no poseen ni admiten sujeto. Este fenómeno típicamente está ausente en idiomas no *pro-drop*; tanto el inglés como el francés requieren el uso de pronombres auxiliares o "pleonásticos" (también conocidos como *dummy pronouns*) como un sujeto sin contenido semántico para verbos impersonales:

- (3-inglés) ***You*** *rest well in this hotel.*
- (3-francés) ***On*** *se repose bien dans cet hôtel.*
- (4-inglés) ***It*** *rained very hard.*
- (4-francés) ***Il*** *a plu très fort.*

La omisión de sujeto y las oraciones impersonales resultan problemáticas para el procesamiento informático de una oración al dificultar el reconocimiento de los argumentos de una acción verbal, además de suponer un problema para la traducción automática entre lenguas *pro-drop* como el español y no *pro-drop* como el inglés. Por estas razones resulta de interés poder detectar automáticamente estos fenómenos en español.

Dado que la mayor parte de los estudios de procesamiento de lenguaje natural se centran en el inglés, un idioma que no presenta estos fenómenos, la detección de sujeto omitido y usos impersonales es un problema menos explorado en la literatura, identificándose solamente un número muy limitado de antecedentes en el área.

1.1. Método

El procesamiento del lenguaje natural (PLN), la aplicación de metodologías y herramientas de las ciencias informáticas al modelado, análisis, comprensión y generación del lenguaje humano, constituye un campo que se ha caracterizado por un gran número de avances en las últimas décadas gracias a la inclusión de desarrollos surgidos de la vanguardia de la inteligencia artificial.

El crecimiento en la capacidad de recopilar vastos volúmenes de texto para la generación de corpus de entrenamiento y de procesarlos en sofisticados modelos de *deep learning* no solo ha conducido a la obtención de mayores niveles de performance sino que también ha dado lugar a la exploración de nuevos enfoques que permiten tanto tratar nuevos problemas como resolver problemas previamente abordados de una forma más eficiente o eficaz.

Uno de estos nóveles enfoques, introducido en el paper *Grammar as a Foreign Language* (Vinyals y cols., 2014), consiste en el abordaje de problemas relacionados al análisis o *parsing* de contenido en un idioma en particular mediante la aplicación de un modelo "secuencia a secuencia" (*sequence to sequence*) concebido originalmente para la traducción automática entre dos o más lenguas naturales, efectivamente equiparando el problema de etiquetar las estructuras gramaticales de un lenguaje natural al problema de la traducción entre idiomas. Este y posteriores trabajos como *Semantics as a Foreign Language* (Stanovsky y Dagan, 2018) aplican desarrollos de la Traducción Automática Neuronal (*Neural Machine Translation* o NMT) como modelos de atención, *transformers* y *decoders* a otras áreas del Procesamiento del Lenguaje Natural utilizando el texto a analizar (típicamente como texto plano, sin etiquetar) en el rol de idioma a traducir y el resultado buscado (por ejemplo un árbol de parsing linealizado) como idioma objetivo. De esta forma, resulta posible entrenar un modelo de traducción en base a un corpus de lenguaje natural y la correspondiente versión preetiquetada de su contenido obteniendo un modelo con la capacidad de generalizar el análisis planteado.

En el presente trabajo, se aplicó este enfoque al reconocimiento de oraciones con sujeto omitido en el idioma español. A diferencia de los problemas abordados en los estudios antes citados (orientados al análisis gramatical y semántico, respectivamente, de oraciones en general), la detección de sujetos omitidos constituye un problema específico, pasible de ser interpretado como un problema de clasificación binaria (sintagmas verbales con sujeto omitido o con sujeto explícito) o, como se explica más adelante, de clasificación en tres categorías (frases con sujeto explícito, con sujeto omitido o frases impersonales). Los resultados de la fase de experimentación permitieron validar la aplicabilidad del enfoque a este tipo de problemas, obteniendo rendimientos comparables a los alcanzados en trabajos anteriores bajo enfoques más tradicionales (Rello, Baeza-Yates, y Mitkov, 2012) (González y Martínez, 2018) (Gerez, Irazusta, y Irace, 2019) con un uso moderado de recursos de hardware.

1.2. Objetivos

Los objetivos generales de este estudio incluyen:

- Reconocer la presencia de sujeto omitido y de verbos impersonales en el idioma español.
- Explorar la viabilidad de aplicar técnicas de traducción automática a la detección de fenómenos puntuales.

Específicamente, este estudio se propone:

- Generar un corpus paralelo (a partir del corpus AnCora en español) que permita tratar la detección de los fenómenos de sujeto omitido y oraciones impersonales como un problema de traducción.

- Elaborar un clasificador basado en el *framework* de traducción automática OpenNMT capaz de identificar verbos con sujeto omitido y verbos impersonales en oraciones en español. Explorar las configuraciones paramétricas que permiten un mayor rendimiento.
- Evaluar el rendimiento obtenido por el clasificador elaborado y compararlo con los resultados obtenidos por trabajos previos en cuanto a clasificación de sujetos en español.
- Evaluar el uso de recursos de cómputo requeridos para el entrenamiento y aplicación del clasificador elaborado.

1.3. Estructura del informe

El resto del documento se estructura de la siguiente manera:

- El capítulo 2 introduce el marco teórico para el proyecto, presenta en mayor detalle los conceptos relevantes para el proyecto en lo respectivo a fenomenología lingüística (sección 2.1) y a técnicas de aprendizaje automático (sección 2.2).
- El capítulo 3 contiene una reseña de antecedentes relevantes para este estudio en la literatura. En particular, se referencia trabajos previos respecto a la aplicación de técnicas de traducción automática a tareas de análisis lingüístico (sección 3.1) y se examinan proyectos anteriores orientados a la detección de sujetos omitidos y verbos impersonales en español (sección 3.2).
- El capítulo 4 comprende el desarrollo del proyecto, detallando las técnicas propuestas.
- El capítulo 5 presenta los resultados experimentales obtenidos durante el transcurso de este trabajo, así como un análisis de los mismos, una comparación con los resultados obtenidos en proyectos anteriores y un estudio de las fuentes de error en los clasificadores elaborados.
- El capítulo 6 expone las conclusiones del estudio, así como posibles líneas de investigación para trabajos futuros.

Se dispone además de anexos que documentan en mayor detalle las configuraciones empleadas y los resultados obtenidos.

Capítulo 2

Conceptos

En esta sección se presentarán conceptos relevantes para el trabajo a modo de marco teórico. Estos incluyen tanto elementos lingüísticos que forman parte del problema a tratar como conceptos vinculados a las técnicas de procesamiento de lenguaje natural (PLN) y aprendizaje automático empleados en este estudio.

2.1. Conceptos lingüísticos

2.1.1. Clases gramaticales y grupos sintácticos

Las palabras en español pueden clasificarse en nueve categorías denominadas clases gramaticales o clases léxicas: (Real Academia Española, 2011)

- **Sustantivos** - palabras que designan objetos y conceptos, como "río" o "arte")
- **Verbos** - palabras referidas a acciones o estados como "comer" o "dormir'
- **Adjetivos** - descripciones que aplican para un sustantivo como "grande" o "verde"
- **Adverbios** - descripciones o contexto que aplica a un verbo o adjetivo, como "bien", "antes" o "perfectamente"
- **Determinantes** - categoría que incluye artículos como "el", "la", posesivos como "mi", "nuestro" y deíticos como "este" o "aquel" entre otros
- **Pronombres** - referencias a una persona u otro sustantivo, como "yo", "aquel" y "dónde"
- **Preposiciones** - palabras que indican el rol de un argumento en una oración, como "de", "con" y "para"
- **Conjunciones** - conectan o contraponen grupos de elementos, como "y", "pero", o "que"

- **Interjecciones** - exclamaciones como “ay” y “¡rayos!”.

En las disciplinas informáticas es común referirse a estas categorías por su nombre en inglés, *part of speech*, abreviado como POS.

La categoría de cada palabra determina su comportamiento sintáctico (la manera en la que se la puede utilizar en los enunciados, en qué estructuras puede aparecer) y su morfología (qué formas puede tomar, como la distinción de número singular-plural en sustantivos o la conjugación en verbos). (Real Academia Española, 2011)

El español muestra cierta flexibilidad respecto a la categoría de ciertas palabras. A modo de ejemplo, “cachorro” puede emplearse tanto como un sustantivo (“El cachorro juega.”) y como adjetivo (“Es un perro cachoro.”) con una semántica (significado) similar. También existen casos de homonimia en los que palabras correspondientes a categorías distintas y sin tener necesariamente un vínculo semántico adoptan la misma forma, como ocurre con la preposición “para” (preposición, “para ti”) y la forma conjugada “para” del verbo “parar”. (Real Academia Española, 2011)

En los enunciados y oraciones, las palabras se agrupan en estructuras dictadas por la gramática del idioma. Las reglas y relaciones que rigen un idioma pueden modelarse a través de múltiples enfoques, siendo el más extendido el de la gramática de constituyentes, modelo adoptado por la Real Academia Española para sus análisis y publicaciones. En este enfoque, las palabras se articulan en grupos sintácticos, también denominados “frases” o “sintagmas”. Cada grupo se encuentra “gobernado” por un núcleo, una palabra que aporta el significado primario de la frase, el cual se modificará o precisará a través de otros elementos (incluyendo potencialmente otros grupos). (Real Academia Española, 2011)

El comportamiento y estructura de un grupo va dictaminado por la clase gramatical de su núcleo. En los “grupos nominales”, el núcleo es un sustantivo o un pronombre que podría especificarse con un determinante o modificarse mediante adjetivos y frases relativas entre otros; son ejemplos de grupos nominales las frases “un **hombre**”, “**Juan**”, “las **niñas** felices” y “el **conductor** que chocó”. En forma similar, se tiene “grupos verbales” (u oraciones) gobernados por un verbo, los cuales pueden incluir grupos nominales que operan como argumentos del verbo (sujeto, objeto directo); son grupos verbales “**ganó**”, “el gato **caza** un ratón”, “**hemos** hecho todo lo que pudimos”). Se puede identificar también grupos adjetivales (gobernados por un adjetivo, “muy **feliz** de haber regresado a salvo”), adverbiales, interjectivos y preposicionales (compuestos por una preposición y un grupo nominal o verbal para el que aplica: “**de** mi amigo”, “**hasta** que termine la obra”). (Real Academia Española, 2011)

2.1.2. Sujeto omitido

En el idioma español, las oraciones tradicionalmente se conciben como una construcción compuesta de dos elementos básicos: el sujeto y el predicado, donde el sujeto corresponde a un grupo sintáctico nominal (gobernado por un sustantivo o pronombre) sobre el cual versa el predicado, un grupo sintáctico verbal

(gobernado por un verbo conjugado). Si bien históricamente ha sido común identificar al sujeto como el elemento que realiza la acción expresada por el verbo, actualmente se reconoce que los sujetos no se caracterizan por un rol semántico (como ser el *agente* de la acción descrita por el verbo) sino que se trata de un concepto puramente sintáctico, es decir, relacionado a la estructura formal de la gramática. En los siguientes ejemplos se marca en negrita el sujeto (Real Academia Española, 2011)

- (5) ***El niño*** *come pescado.* (sujeto opera como agente del verbo "comer")
- (6) ***Ella*** *merece ese puesto.* (en verbos como "merecer", "gustar" o "caber" se identifica un sujeto aunque el mismo no opera como agente a nivel semántico)
- (7) ***El barco*** *fue avistado.* (en las construcciones pasivas, el sujeto corresponde al paciente de una acción en vez de a su agente)
- (8) *Lo digo* ***yo***. (ejemplo de que el sujeto no necesariamente precede al verbo)

Al igual que la mayor parte de las lenguas indoeuropeas, familia lingüística que incluye al español, nuestro idioma presenta concordancia entre las formas finitas (conjugadas) de un verbo y su sujeto. Este fenómeno, expresado como la necesidad de conjugar los verbos según el número (singular o plural) y persona gramatical del sujeto contribuye a la identificación del sujeto, aunque no necesariamente de forma inequívoca. (Real Academia Española, 2011)

Otro fenómeno sintáctico que ayuda a identificar los componentes de una oración (incluyendo la determinación de su sujeto) es el orden en los que estos aparecen. Si bien el español es relativamente flexible respecto a su orden oracional (sobre todo en comparación a lenguas como el inglés, el francés y el mandarín), existe una fuerte tendencia a ubicar el sujeto en posición anterior al verbo, típicamente en una estructura sujeto-verbo-objeto (SVO). De esta forma, una oración como *"La pala rompió la piedra"* típicamente se interpretará según la estructura SVO, entendiendo a "la pala" como el sujeto que causó la rotura de la piedra, si bien en un principio la interpretación contraria (la piedra como sujeto que causa la rotura de la pala) es válida. Aunque asumir que el grupo nominal que precede al verbo corresponde a su sujeto es una heurística válida y relativamente efectiva, no es infalible. Podemos igualmente construir oraciones donde la interpretación más probable requiere identificar un sujeto posterior al verbo (sujeto pospuesto) como es el caso de *"La manzana eligió Juan"* donde hay tanto elementos semánticos (es más lógico que una persona escoja una fruta que viceversa) como sintácticos (si Juan fuera el objeto directo y no el sujeto, sería esperable que estuviera antecedido por la preposición "a", *"La manzana eligió a Juan"*, al ser un complemento directo animado) para identificar que "Juan" es el sujeto pese a aparecer pospuesto al verbo. Los sujetos pospuestos se hacen más comunes en oraciones con verbos en modo imperativo (*"Atendelo vos"*), subjuntivo (*"¡Que lo haga él!"*) y en preguntas (quizás por influencia

del francés, *"¿Será él el indicado?"*). A nivel coloquial, los sujetos pospuestos resultan también especialmente comunes en el español rioplatense (*"Tremenda bici se compró el gurí"*). (Kornfield y Kuguel, 2012)

La gramática del español permite formular oraciones que carecen de un sujeto explícito como en *"Siempre hablan mucho"*, donde la conjugación del verbo permite inferir la existencia de un sujeto plural de tercera persona ("ellos" o "ellas") que no está presente en la frase. Este fenómeno, tradicionalmente denominado "sujeto tácito", se entiende bajo un modelo de gramática generativa no como una ausencia de sujeto sino como una omisión. La *Nueva gramática básica* adscribe a estas líneas al describir estos sujetos como "sujetos que carecen de expresión fónica". (Real Academia Española, 2011) A nivel teórico, según el marco "Principios y Parámetros" de la gramática generativa chomskyana, responde a que el español posee el parámetro de sujeto nulo o *pro-drop*. En este aspecto, el español contrasta con otras lenguas europeas como el inglés y el francés que no presentan omisión por lo que una frase como *"Siempre hablan mucho"* necesariamente incorporará explícitamente un sujeto ("***They*** *speak a lot*", "***Ils*** *parlent beaucoup*"). Debe notarse, sin embargo, que el rasgo *pro-drop* tiende a ser la norma a nivel mundial incluso para lenguas como el mandarín que no presentan concordancia verbal entre verbo y sujeto; la obligatoriedad del sujeto observada en el inglés y el francés encontrada solo en determinadas regiones (incluyendo el centro y norte de Europa). (Suárez Palma, 2012) (Real Academia Española, 2011) (Haspelmath, 2001) (Dryer, 2013)

Con el fin de adherirse al precedente sentado por (González y Martínez, 2018), en el presente trabajo estos sujetos "tácitos" se referirán como **sujetos omitidos**.

Puede observarse que la determinación de si una oración dada presenta sujeto explícito u omitido puede presentar ambigüedades. A modo de ejemplo, la oración *"Cocinaron las milanesas."* podría corresponder tanto a una oración con sujeto omitido (donde un grupo de personas no mencionadas cocinan el plato denominado "milanesa") o a una oración con sujeto explícito pospuesto (las milanesas, mujeres de la ciudad de Milán, cocinaron un plato no especificado).

2.1.3. Verbos y oraciones impersonales

Se puede identificar dos variedades de oraciones que no presentan un sujeto explícito: aquellas en las que se sobreentiende la existencia de un sujeto y que podrían reexpresarse incluyendo un grupo nominal que adopte tal rol (por ejemplo *"Toma agua"* puede cambiarse por *"Él toma agua"*) y aquellas en las que no resulta gramaticalmente válido incluir un sujeto, como es el caso de *"Hoy llueve"*, *"Habrá buena cosecha"* y *"Se trabaja mucho"*. (Real Academia Española, 2011)

Las oraciones impersonales pueden deber su condición a un atributo propio del verbo empleado, como ocurre en los llamados *verbos impersonales léxicos* (entre los que se comprenden los verbos meteorológicos como "llover", "relampaguear" o "amanecer") cuya semántica excluye la posibilidad de identificar un sujeto, o, por el contrario, pueden contener construcciones impersonales con ver-

bos que en otros contextos sí admiten un sujeto, como en la frase *"En tren, se viaja más cómodo"* donde no resulta posible definir un sujeto para el verbo "viajar" que usualmente sí lo tiene. (Gerez y cols., 2019) (Real Academia Española, 2011)

La *Nueva gramática básica* de la RAE define los siguientes casos de oraciones impersonales: (Real Academia Española, 2011)

- Las definidas por verbos impersonales léxicos relacionados a fenómenos atmosféricos: *"Nevó en París"*.
- Oraciones impersonales con el verbo haber, referidas a la existencia de un objeto (que, sintácticamente, opera como complemento y no sujeto): *"Hay vacantes"*.
- Oraciones impersonales con formas del verbo "ser", "estar", "caber" o "hacer": *"Es de noche"*.
- Ciertos usos de verbos como "dar", "ir", "bastar", entre otros: *"Se le dio por hacerlo"*, *"¿Cómo te va?"*, *"Basta con apretar un botón"*.
- Oraciones con un sujeto inespecífico: *"Llaman a la puerta"*.
- Oraciones impersonales introducidas con el clítico "se": *"Se duerme bien en este hotel"*. Cabe distinguir este caso de las oraciones reflejas como *"Se duerme tarde"* o *"se critican mutuamente"* donde el pronombre "se" indica una reflexividad de la acción que, sin embargo, preserva un sujeto identificable. Un elemento que permite distinguir entre estos dos usos de "se" está en que el segundo uso (que sí permite sujetos) admite la sustitución del clítico "se" por otros pronombres como "me" y "nos": resulta admisible decir *"me duermo tarde"* o *"nos criticamos mutuamente"* pero no *"*me duerme bien en este hotel"*.

Si bien desde un punto de vista puramente sintáctico las oraciones impersonales pueden verse como un caso obligatorio de sujeto omitido o *pro-drop* (Suárez Palma, 2012), es posible apreciar una diferencia cualitativa entre una oración con un sujeto tácito que podría adicionarse un sujeto expreso y aquellas en las que esto no es permisible. La capacidad de un modelo de distinguir entre oraciones impersonales y oraciones con sujeto omitido tiene además usos prácticos, ya que únicamente en las últimas resulta posible reconstruir el sujeto, lo cual puede ser necesario para analizar la semántica de un texto o para traducirlo a un idioma sin el parámetro *pro-drop*. Por este motivo, en trabajos anteriores enfocados en la detección de sujetos omitidos en el español se trata a la oración impersonal como una tercera posibilidad, enfoque que fue también adoptado por el presente trabajo. (González y Martínez, 2018) (Gerez y cols., 2019)

2.1.4. Verbos finitos y no finitos

La morfología del español permite que un verbo adopte un número de formas o conjugaciones, las cuales pueden clasificarse en formas personales o "finitas" y formas no personales o "no finitas". Las **formas finitas** son aquellas para las que se puede reconocer tiempo verbal, aspecto, modo y, principalmente, la persona y número de su sujeto, como en los siguientes ejemplos:

- (9) *El hombre* ***toma*** *café.*
- (10) ***Fueron*** *en auto.*
- (11) *¡****Hacelo*** *vos!*

Las formas finitas se caracterizan por poder formar oraciones por sí mismas, sin necesidad de operar junto a un verbo auxiliar. Las **formas no finitas**, por el contrario, no pueden constituir oraciones salvo con el apoyo de verbos auxiliares. El español presenta tres variedades de formas no finitas: el infinitivo (como "cantar", "comer" o "partir"), el gerundio ("cantando", "comiendo", "partiendo") y el participio ("cantado", "comido", "partido"). (Real Academia Española, 2011) Puede observarse que estas formas verbales no tienen la capacidad de formar oraciones salvo utilizando verbos auxiliares aunque pueden utilizarse en otros roles, por ejemplo como un argumento de otro verbo (donde la forma no verbal opera como sustantivo o, más precisamente, como un grupo sintáctico nominal) o como adjetivos.

- (12) ***Tomar*** *café.* (no es una oración completa)
- (13) *Él* ***va a tomar*** *café.* (oración completa, el infinitivo se utiliza en conjunto con el verbo auxiliar)
- (14) *Le gusta* ***tomar*** *café.* (el verbo en infinitivo forma parte de un grupo nominal, "tomar café", que opera como objeto del verbo finito "gusta")

Tradicionalmente se ha definido la diferencia entre formas finitas y no finitas en relación a la presencia o ausencia de marcas de concordancia personal y de tiempo verbal, lo cual resulta apropiado para el español aunque resulta inconsistente para formas análogas en otras lenguas (a modo de ejemplo, el portugués admite "infinitivos personales" que presentan concordancia personal como *falarmos*, "hablar (nosotros)", mientras que el latín presenta infinitivos correspondientes a tiempo pasado o futuro). Actualmente se favorece definir los verbos finitos como aquellos que pueden servir como núcleo de una oración declarativa, mientras que los verbos no finitos únicamente pueden aparecer en carácter de complementos o con un rol nominal (como ocurre con el infinitivo en oraciones como "yo quiero *cantar*") o atributivo (adjetival o adverbial, como el participio en "*terminado* el año" o el gerundio en "*esforzándose* lo consiguió"). (Matthews y Matthews, 1997)

Las tres formas verbales no finitas del español se emplean en construcciones verbales que utilizan verbos auxiliares para indicar características gramaticales

de tiempo, aspecto y voz (como el tiempo futuro en “voy a *cantar*“, el aspecto progresivo en “está *comiendo*” o la voz pasiva en “fue *partido*”) así como en construcciones donde operan como complemento a un verbo principal como en ‘cree *cantar* muy bien‘. Resulta relevante observar que estos casos existe un único verbo finito en la construcción, incluso en construcciones complejas como “quisiera haber estado estudiando”, por lo que la cuestión de si se define o no el sujeto de una construcción verbal en forma explícita puede expresarse en términos de si existe un sujeto explícito para el verbo finito que preside la construcción. De esta forma, en la oración “Va a haber cocinado muy bien” puede reconocerse un sujeto omitido que “va a haber cocinado” como el sujeto implícito del verbo “va” (forma finita en presente y tercera persona singular del verbo “ir”), pese a que a nivel semántico el rol que cumplirá dicho sujeto es cocinar (referenciado en la construcción por la forma no finita “cocinado”, un participio). (Real Academia Española, 2011)

En algunos casos resulta posible identificar a nivel semántico a un actor que ocuparía el rol de sujeto para un verbo no finito (por ejemplo en “partimos a primera hora para *llegar* temprano”, puede identificarse que la acción de “llegar” referirá a “nosotros”) pero la misma no tiene implicancias a nivel sintáctico por lo que no resulta apropiado considerarlo para el problema de la detección de sujetos omitidos. Por estos motivos, en este proyecto únicamente se considerará la evaluación de si un verbo posee o no un sujeto omitido para verbos finitos. (Real Academia Española, 2011)

2.2. Conceptos de aprendizaje automático y PLN

2.2.1. Procesamiento de lenguaje natural

Se conoce como “procesamiento de lenguaje natural” (PLN, en inglés *Natural Language Processing* o NLP) a la disciplina de las ciencias de la computación enfocada en el manejo informático del lenguaje natural, es decir, los idiomas empleados cotidianamente por los seres humanos como el inglés y el español, ya sea en forma hablada o escrita, en contraposición a los lenguajes formales como los lenguajes de programación o los formatos de datos estructurados. Por su naturaleza, se trata de un campo interdisciplinario con componentes de lingüística e informática. La disciplina contempla problemas como el reconocimiento del significado (semántica) de un texto, la traducción automática entre idiomas, la generación de lenguaje natural, entre otros.

El lenguaje natural presenta múltiples desafíos para su tratamiento por parte de sistemas informáticos al presentar una estructura variable y propensa a la ambigüedad en contraposición a los modelos estructurados y generalmente libres de ambigüedad que se encuentran en los datos numéricos, lenguajes de programación y otros formalismos. Las lenguas humanas presentan además una gran diversidad, por lo que los avances en el procesamiento de contenidos en un idioma no necesariamente resultarán aplicables para otro idioma.

Tareas comprendidas en PLN

El área del procesamiento del lenguaje natural trata diversos problemas que suponen la extracción de información manipulable por sistemas informáticos a partir de texto en lenguaje natural y/o la generación del mismo.

Entre las tareas más características de la disciplina se encuentran tareas de "bajo nivel", usualmente orientadas al reconocimiento de estructuras morfológicas y sintácticas para construir una representación del idioma con la que otros desarrollos puedan operar, y tareas de "alto nivel", las cuales comprenden consideraciones respecto a la interpretación o semántica de los textos.

Dentro de las tareas de bajo nivel se destacan el etiquetado de categoría gramatical (*POS-tagging*, clasificar cada palabra en sustantivos, adjetivos, verbos, pronombres, etc.) y el análisis sintáctico o *parsing*, el cual consiste en identificar la estructura de una oración y las relaciones sintácticas entre sus componentes. La detección de sujeto omitido puede considerarse asimismo como una tarea de bajo nivel.

Por su parte, las tareas de alto nivel comprenden, entre otros, la generación de resúmenes automáticos, la traducción automática y la generación de respuestas en lenguaje natural.

Enfoques en PLN

Históricamente se han empleado dos grandes enfoques en PLN: sistemas basados en reglas y sistemas basados en aprendizaje automático, aunque existen también proyectos con un enfoque híbrido.

En los sistemas basados en reglas se parte de conocimiento previo (*a priori*) sobre la estructura de la lengua a procesar. Dichos sistemas se basan en modelos de gramática desarrollados por lingüistas que buscan identificar una estructura formal dentro del lenguaje natural. El modelado de una oración en el marco de esta clase de puede adoptar la forma de un árbol de *parsing* y, por regla general, no resulta inequívoco ya que una misma oración podría admitir múltiples representaciones como se muestra en la figura 2.1; la determinación de qué representación corresponde asignar a una oración dada puede resolverse mediante modelos probabilísticos.

Los sistemas basados en reglas se encuentran limitados por la flexibilidad del modelo lingüístico adoptado. A modo de ejemplo, si el modelo partiera de la base de que las oraciones en español siguen una estructura sujeto-verbo-objeto como en "El niño (S) come (V) la manzana (O)", el sistema sería incapaz de procesar correctamente desviaciones a dicho modelo que pueden ocurrir en el habla coloquial como "Come la manzana, el niño". Esta clase de sistemas también presenta la desventaja de encontrarse fuertemente ligados a la gramática de una lengua en particular, por lo que los desarrollos para tratar texto en un idioma podrían resultar ineficaces para procesar contenido en un idioma distinto.

Por otro lado, los sistemas basados en aprendizaje automático no necesariamente asumen una estructura conocida en el contenido a procesar sino que construyen un modelo a partir de un conjunto de enunciados de ejemplo deno-

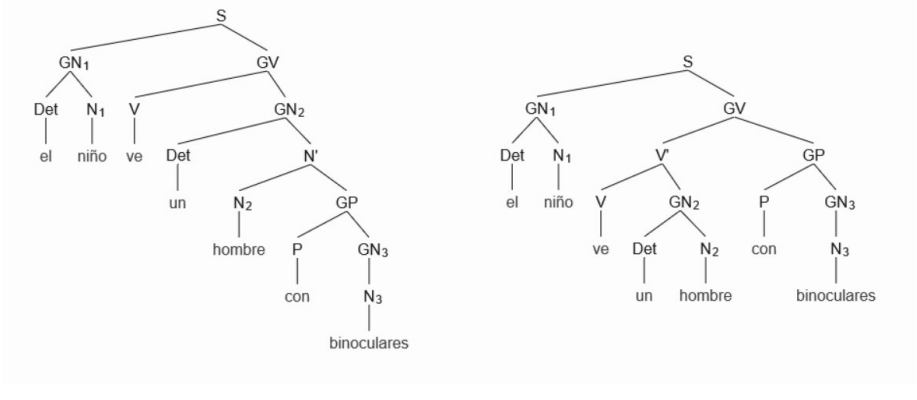

Figura 2.1: Dos análisis posibles de la oración "El niño ve un hombre con binoculares" en un ejemplo de gramática de constituyentes. El árbol de *parsing* a la izquierda asume que "con binoculares" describe al hombre que el niño ve, mientras que el árbol a la derecha asume que "con binoculares" describe la manera en la que el niño ve al hombre. La ambigüedad respecto a qué elemento se referencia en una frase preposicional (*Prepositional Phrase Attachment*) es un problema habitual en el procesamiento de idiomas indoeuropeos como el inglés y el español.

minado *corpus*, el cual usualmente es extenso (de miles a millones de oraciones) para que sea representativo de la diversidad de construcciones gramaticales y léxicas que pueden hallarse en el lenguaje natural. Dado que la estructura del modelo se basa en patrones detectados en los datos de entrenamiento en vez de en supuestos previos sobre la estructura del idioma se puede decir que este enfoque maneja conocimiento *a posteriori*. En las últimas dos décadas, los enfoques basados en aprendizaje automático han tomado una particular relevancia en PLN obteniendo resultados del estado del arte para la mayor parte de los problemas tratados por la disciplina; esto puede atribuirse en parte a desarrollos teóricos como nuevas variantes de redes neuronales y sistemas de atención así como a la mayor disponibilidad de textos a utilizar como corpus de entrenamiento a partir de textos obtenidos de internet y la mayor capacidad de procesamiento (en particular con el desarrollo de unidades de procesamiento paralelo como GPUs y TPUs) que han permitido el entrenamiento de modelos más sofisticados basados en una mayor cantidad de datos.

Los sistemas basados en aprendizaje automático tienden a presentar una mayor flexibilidad para adaptarse a otros lenguajes naturales. Dado que las características gramaticales y el vocabulario del idioma no forman una parte intrínseca del modelo (como ocurre en los sistemas basados en reglas) sino que se los abstrae del corpus de entrenamiento, muchos modelos pueden adaptarse a un idioma diferente al utilizar un corpus en un idioma distinto, aunque esto dependerá de la disponibilidad de un corpus de las características necesarias en el

idioma deseado. En tiempos recientes se ha logrado avances en el entrenamiento de modelos intrínsecamente multilingües como es el caso de ChatGPT (2022), un modelo de generación de lenguaje natural capaz de producir texto en más de 90 idiomas naturales. (Christensen, 2023)

2.2.2. Aprendizaje automático

El aprendizaje automático o *Machine Learning* constituye un campo de investigación enmarcado dentro de la inteligencia artifical (IA) basado en el reconocimiento de patrones que se infieren de un conjunto de ejemplos o "datos de entrenamiento". Estos datos se emplean para refinar un modelo candidato a solucionar un problema dado, empleando métodos de optimización estadísticos que podrían verse como análogos al modo en que un ser humano es capaz de aprender a reconocer un patrón a partir de ejemplos. En un caso de aprendizaje automático exitoso, el modelo resultará capaz de *generalizar* los patrones presentes en los ejemplos de entrenamiento de forma que genere salidas válidas al presentarle ejemplos análogos no incluidos en el conjunto de datos de entrenamiento. (Norvig, 2020)

Aprendizaje supervisado

Los algoritmos de aprendizaje automático pueden categorizarse como métodos de aprendizaje *supervisado* y de aprendizaje *no supervisado*. En un aprendizaje supervisado los datos utilizados para el entrenamiento especifican tanto la entrada que recibirá el modelo como la salida esperada. En la terminología del campo, se dice que los datos empleados en este tipo de aprendizaje se encuentran "preetiquetados", un término derivado de problemas de clasificación en los que el objetivo del modelo es usualmente el de asignar a cada entrada la "etiqueta" de la categoría que le corresponde. (IBM, 2020a)

Un ejemplo clásico de aprendizaje automático supervisado aplicado a PLN es la categorización de comentarios o reseñas de un producto o servicio como "positivos", "neutros" o "negativos". En dicho caso, los datos de aprendizaje se componen de duplas que contienen el texto del comentario (la entrada del clasificador) y la categoría que se espera detectar (la salida esperada). A partir de estas duplas, el modelo de aprendizaje automático deberá ser capaz de identificar patrones que resulten relevantes para la clasificación, como que la palabra "excelente" tenderá a aparecer en comentarios positivos mientras que "mal" será más común en comentarios negativos (aunque también se deberá identificar patrones más complejos, como que "nada mal" tiene una connotación positiva).

En estos sistemas de aprendizaje automático se puede identificar tres componentes (ilustrados en la figura 2.2): (School of Information Berkeley, 2022)

- Un **proceso de decisión** que genera una posible solución para el problema a tratar (que podrá ser correcta o no) en base a la entrada recibida y un conjunto de parámetros ajustables del modelo.

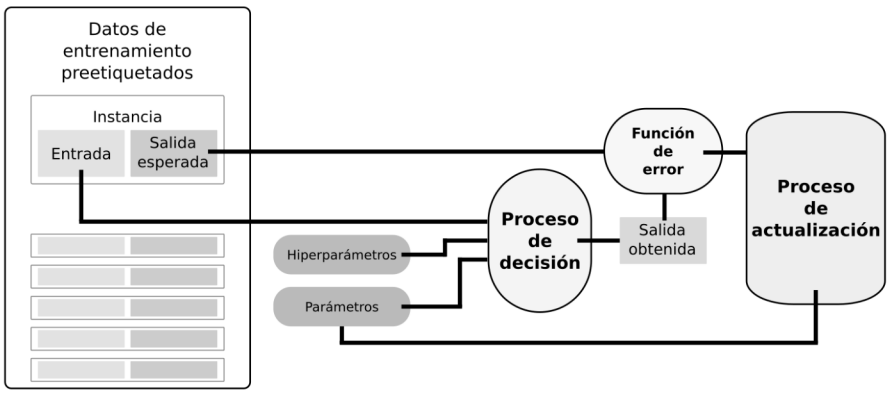

Figura 2.2: Esquema de los componentes de un sistema de aprendizaje automático supervisado.

- Una **función de error** que opera como una métrica de la diferencia entre la salida esperada en los datos de entrenamiento y la generada en un momento dado por el proceso de decisión. Esta función típicamente tendrá un resultado de 0 en caso de que ambas salidas coincidan o de un valor positivo en caso contrario. Según el problema a resolver se podría tener una función binaria (0 para salida correcta, 1 para incorrecta) o admitir múltiples valores positivos según una valoración de cuánto se aleja la salida obtenida de la salida esperada (por ejemplo el valor absoluto de su diferencia en caso de tratarse de valores numéricos).
- Un **proceso de actualización** u **optimización** que ajusta los parámetros del proceso de decisión para minimizar el error medido por la función de error.

El entrenamiento de un sistema de aprendizaje automático puede verse, por lo tanto, como un problema de optimización: encontrar los parámetros del proceso de decisión que minimicen el resultado de la función de error para los ejemplos contemplados en los datos de entrenamiento. Este problema se trata a través de diversos tipos de procesos de actualización, los cuales suelen ser iterativos: cada ejemplo o grupo de ejemplos se procesa en una fase de entrenamiento que resulta en un ajuste a los parámetros a utilizar en la siguiente iteración del entrenamiento hasta que se alcance un criterio de finalización, como alcanzar una cantidad predefinida de iteraciones, conseguir un determinado rendimiento o constatar que la disminución del error respecto a la iteración precedente cayó por debajo de un umbral que sugiere que realizar más iteraciones no resultará productivo. (School of Information Berkeley, 2022) (Norvig, 2020)

Además de los parámetros que pueden ajustarse a través del proceso de actualización, los modelos empleados en el proceso de decisión presentan características que no se contempla modificar durante su entrenamiento al conside-

rarse elementos intrínsecos a la familia de modelos que se toma como universo para el problema de optimización. Estas características se denominan hiperparámetros y pueden incluir elementos como la dimensionalidad del modelo, los formatos empleados para la entra y la salida, los criterios empleados durante el proceso de actualización, criterios de finalización, etc. Un estudio para abordar un problema mediante un proceso de aprendizaje automático usualmente comprenderá la investigación de múltiples conjuntos de hiperparámetros para definir qué tipo de modelos se presenta más promisorio para tratar el problema a resolver. (Nyuytiymbiy, 2020)

Se aprecia una desventaja del aprendizaje supervisado en la necesidad de determinar la salida esperada al crear el conjunto de datos de entrenamiento, lo cual podría requerir un etiquetado manual por parte de seres humanos, un proceso potencialmente lento, costoso y pasible de errores que podrían impactar en el resultado del sistema entrenado. Esto resulta particularmente problemático al apreciarse que la calidad de los resultados de muchos algoritmos de aprendizaje automático muestra una dependencia directa respecto al número de ejemplos empleados en su entrenamiento. Una práctica frecuente consiste en emplear un conjunto de datos preetiquetados manualmente por un grupo de expertos considerado plenamente confiable (denominado "estándar de oro" o *gold standard*) y utilizar un método de preetiquetado automático preexistente con un menor grado de confianza para generar un mayor número de ejemplos, los cuales podrían ponderarse con una menor relevancia a los contenidos en el estándar de oro. (IBM, 2020a)

Aprendizaje no supervisado

En un aprendizaje no supervisado el modelo no cuenta con datos preetiquetados sino que el algoritmo busca identificar patrones y relaciones sin una guía generada por humanos de cuál será el resultado esperado. Un uso típico de esta clase de modelos se encuentra en los problemas de agrupamiento o *clustering*, en los que se desea reconocer la estructura interna de un grupo de elementos. A modo de ejemplo, un aprendizaje no supervisado podría tomar como entrada datos sobre especímenes botánicos o zoológicos y presentar un agrupamiento que ayude a los investigadores a determinar a qué especies corresponden. Se puede encontrar un uso habitual de este tipo de modelos en los sistemas de recomendaciones: un modelo de aprendizaje automático podría reconocer patrones en los productos adquiridos por los clientes de una plataforma de comercio electrónico o en los videos visualizados por un en una plataforma de streaming para determinar los elementos más relevantes que se podrán presentar al usuario como recomendaciones.(Delua, 2021)

En los aprendizajes no supervisados también pueden reconocerse los elementos de un problema de optimización: se tiene un proceso que transforma la entrada en una determinada salida (usualmente una representación de la estructura interna, como un vector de n dimensiones cuya distancia a los vectores generados por otras entradas es representativa del grado de similitud entre las mismas) con parámetros ajustables e hiperparámetros intrínsecos al modelo,

una función de error que podría estar definida por el grado de dispersión de los elementos u otra característica que interese minimizar y un proceso de actualización a través del cual los parámetros del modelo puedan ajustarse para lograr una mejor calidad en las salidas. (Delua, 2021)

Al no requerir el preetiquetado de datos, a menudo resulta posible entrenar modelos de aprendizaje no supervisado con conjuntos de datos superiores por varios órdenes de magnitud a los utilizados en modelos de aprendizaje supervisado; en PLN existen múltiples instancias de corpus de entrenamiento con miles de millones de palabras obtenidos de datos publicados en internet que pueden utilizarse para formar modelos lingüísticos no supervisados pero cuyo preetiquetado para modelos supervisados resultaría prohibitivamente costoso. (Delua, 2021)

Generalización, *overfitting* y segmentación de datos de entrenamiento

La efectividad de un modelo de aprendizaje automático reside en su capacidad de inferir patrones relevantes al problema a tratar que puedan reconocerse en instancias por fuera de las utilizadas durante su entrenamiento. Cuando un modelo presenta esta característica se dice que *generaliza*.

La capacidad de generalización buscada en los modelos en algunos casos puede verse en conflicto con el problema de optimización empleado durante el entrenamiento ya que el proceso de actualización podría explotar patrones espurios presentes únicamente en los datos de entrenamiento que no generalizan a otras instancias, un fenómeno conocido como sobreajuste u *overfitting*. En un caso extremo, un modelo podría tomar todas las características brindadas para cada ejemplo (independientemente de su relevancia al problema tratado) y brindar el resultado asociado ante una coincidencia perfecta, efectivamente memorizando la etiqueta a asignar para cada ejemplo pero sin explotar los patrones emergentes de combinar las características relevantes para el problema en general. Esta técnica lograría minimizar la función de error para todas las instancias del entrenamiento pese a que el modelo no tendría la capacidad de generalizar para las instancias "no memorizadas" no incluidas en el conjunto de entrenamiento. (IBM, 2020b)

Para disminuir el efecto de este problema, los proyectos de aprendizaje automático tienen como práctica establecida la segmentación de los datos disponibles para el entrenamiento del modelo en al menos dos subconjuntos: un conjunto para utilizar efectivamente en el entrenamiento y un conjunto reservado para evaluación. El propósito de esta división es contar con un modo de evaluar el rendimiento de un modelo ante instancias que no afectan directamente el comportamiento del modelo, de modo que el resultado permita evaluar el aprendizaje generalizado ignorando el efecto de los patrones espurios. (IBM, 2020b)

En particular, si el resultado de evaluar un modelo con el conjunto de evaluación empeora tras un número de instancias de entrenamiento puede deducirse que el sistema está experimentando *overfitting*. En estos casos es una práctica común detener el entrenamiento, un criterio conocido como detención temprana

o *early stopping*. (IBM, 2020b)

Si un proyecto de aprendizaje automático incluye la exploración de múltiples combinaciones de hiperparámetros es posible encontrar un problema análago: las combinaciones de hiperparámetros que permiten la obtención de mejores resultados con el conjunto de datos de evaluación podrían estar explotando patrones espurios en este segundo subconjunto. En estos casos resulta de utilidad emplear tres segmentos de datos: uno de entrenamiento utilizado por el proceso de actualización en un modelo con hiperparámetros dados, uno de desarrollo utilizado para evaluar los resultados de diferentes modelos con variaciones en sus hiperparámetros y un conjunto de evaluación para evaluar sin sesgos el rendimiento de la solución candidata tras la exploración de hiperparámetros. Esta clase de segmentación en tres subconjuntos resulta usual en los corpus de texto empleados para problemas de PLN, como se detallará más adelante en la sección *Partición de corpus*. (IBM, 2020b) (Brownlee, 2017)

Algoritmos de aprendizaje automático

El aprendizaje automático puede seguir diversas estructuras y algoritmos que determinan el modo en el que el *proceso de decisión* genera una salida, qué parámetros e hiperparámetros se utilizan y de qué manera el *proceso de actualización* ajustará los parámetros con el objetivo de minimizar la *función de error*. (Norvig, 2020)

Muchos de los primeros sistemas de aprendizaje automático orientados a problemas de clasificación se basaban en **regresión lineal** y en **regresión logística**. En estos métodos se parte de una entrada representada como un vector n dimensiones y se asume que cada categoría puede aproximarse con la partición del espacio n-dimensional mediante fronteras expresables como funciones matemáticas simples, con una estructura conocida y parámetros que pueden calcularse mediante técnicas como la aproximación por mínimos cuadrados. Estos métodos son capaces de lograr resultados prometedores en problemas de categorización simple, incluyendo la detección de correos como spam, aunque la *capacidad expresiva* de esta clase de modelos (la cantidad y tipo de problemas que son capaces de modelar adecuadamente) es limitada y no resulta aplicable para problemas complejos. (School of Information Berkeley, 2022) (Norvig, 2020)

Otra familia de algoritmos con una mayor capacidad expresiva se encuentra en los clasificadores probabilísticos, los cuales buscan generar un modelo para calcular la probabilidad de cada salida posible a partir de una entrada dada de manera que el modelo generado se ajuste lo mejor posible a la realidad identificada en los datos de entrenamiento: se busca un modelo con una estructura dada que maximice la probabilidad de que las entradas observadas correspondan al resultado observado. La variante más extendida de estos clasificadores es el clasificador bayesiano ingenuo (conocido en inglés como *Naïve Bayes Classifier*, NBC) el cual se basa en el supuesto de que cada atributo de una entrada (entradas de un vector o matriz, rasgos representativos de un elemento a modelar, etc) tiene una distribución independiente. Si bien dicho supuesto generalmente no

se cumple (los atributos de una entrada suelen encontrarse interrelacionados), un NBC suele presentar un rendimiento aceptable y muestra una baja complejidad computacional al poder utilizar el teorema de Bayes para calcular la probabilidad condicional de eventos que se presumen independientes. (Gandhi, 2018)

Si bien los métodos basados en regresión y clasificadores probabilísticos continúan vigentes, en las últimas décadas las discusiones sobre aprendizaje automático se han visto dominadas por las redes neuronales, una familia de métodos de aprendizaje automáticos basados en unidades que pueden operar en conjunto para lograr mayores niveles de abstracción y capacidad expresiva.

2.2.3. Redes neuronales

Las redes neuronales constituyen una familia de métodos de aprendizaje automático conformados por unidades con un comportamiento inspirado por el de las redes de neuronas biológicas, de las cuales estos modelos toman el nombre. Las redes neuronales artificiales presentan conexiones entre las unidades (análogas a las sinapsis biológicas) que se ponderan mediante parámetros del modelo. En su forma más básica, cada unidad de una red neuronal toma los valores de un número de entradas (las cuales pueden corresponder a la entrada del modelo o a la salida de otras unidades), los multiplica por la ponderación correspondiente a cada conexión (las cuales pueden ser positivas o negativas) y procesa la suma de estos "estímulos" mediante una función no lineal (denominada "función de activación") cuyo resultado se presenta como salida de la unidad, la cual, a su vez, podrá formar parte de la salida del modelo o utilizarse como entrada para otras unidades.

El proceso de entrenar una red neuronal involucra ajustar las ponderaciones de cada conexión de manera de minimizar el error entre las salidas esperadas y las salidas obtenidas. Esto lleva a que el número de parámetros a entrenar dependa del número de unidades y conexiones. Entre los hiperparámetros aplicables a las redes neuronales se encuentra el número de unidades a emplear, su disposición (qué conexiones se admitirá entre ellas), la dimensionalidad de su entrada y salida, el tipo de función de activación a emplear (siendo las alternativas más frecuentes la arcotangente, una función logística o "sigmoide" y ReLU, que combina una función lineal simple $f(x) = mx$ para valores positivos y $f(x) = 0$ para valores negativos) así como variables relativas al procedimiento empleado en la función de actualización.

Dado que se requiere un parámetro de ponderación para cada entrada de una unidad y que el número de parámetros impacta directamente en el costo computacional del entrenamiento, emplear una arquitectura en la que cada unidad está conectada a todas las unidades restantes resulta en un proceso de entrenamiento computacionalmente costoso ya que el número de parámetros escalaría forma cuadrática con el número de unidades (n^2 conexiones para n unidades). Debido a esto, las arquitecturas de redes neuronales más extendidas organizan las unidades en capas donde cada unidad únicamente procesa la salida de las unidades de la capa anterior, como se ilustra en la figura 2.3 En estos

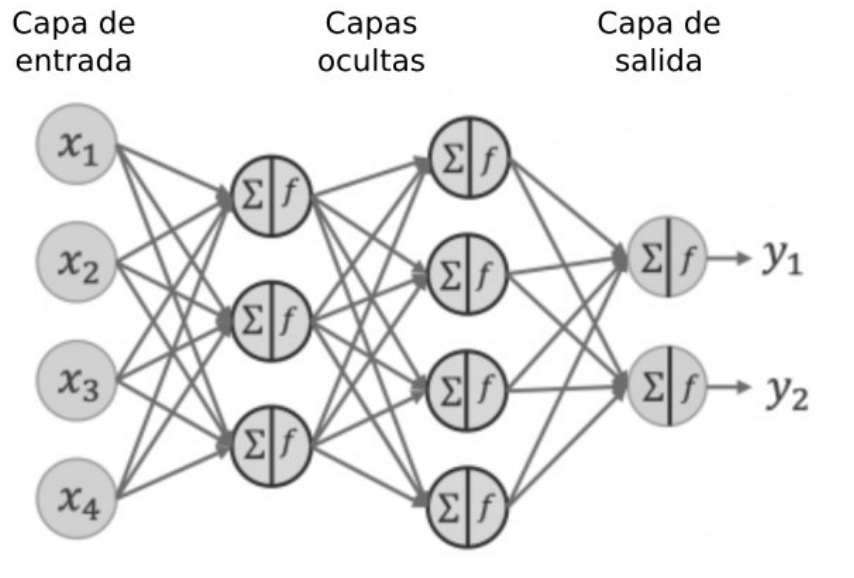

Figura 2.3: Esquema de una red neuronal no recurrente con cuatro capas: una capa de entrada, dos capas ocultas y una capa de salida. Cada unidad solamente procesa como entrada a la salida de las unidades de la capa precedente, cuyos estímulos se ponderan de acuerdo a los parámetros del modelo y luego se someten a una función de activación f no lineal. Diagrama tomado de Melcher (2021).

modelos se tiene una capa de entrada (*input layer*), un número de capas intermedias (usualmente denominadas "capas ocultas" o *hidden layers*) y una capa de salida (*output layer*): las unidades de la *input layer* procesan la entrada recibida por el modelo generando resultados parciales que se toman como insumo para la capa siguiente hasta llegar a la capa de salida, la cual genera los valores devueltos por el modelo. En estas arquitecturas, cada capa contribuye un nuevo nivel de abstracción, por lo que las redes con múltiples capas ocultas presentan una mayor capacidad de reconocimiento de patrones complejos, dando lugar al "aprendizaje profundo" o *Deep Learning*.

Las redes neuronales que siguen el esquema de múltiples capas en los que las unidades únicamente tienen acceso a los resultados de la capa precedente (o de la entrada del modelo en el caso de la *input layer*) se conocen como redes neuronales prealimentadas o *feedforward neural networks* (FFNN). Las últimas dos décadas han presentado considerables desarrollos teóricos con la introducción de nuevas variantes y tipologías de redes neuronales, incluyendo las redes neuronales recurrentes, las cuales se explorarán en mayor detalle más adelante. Estas innovaciones teóricas, combinadas con avances en materia de

hardware que han hecho factible el entrenamiento de modelos con un mayor número de unidades, han posibilitado el desarrollo de modelos con una muy alta capacidad expresiva capaz de dar tratamiento en forma efectiva a problemas que anteriormente no estaban al alcance del campo, como la generación de lenguaje natural realista con una noción adecuada de contexto mediante ChatGPT o la generación de contenido gráfico y de video. (Masood, Nawaz, Malik, Javed, y Irtaza, 2021)

Entrenamiento de redes neuronales por *backpropagation*

El ajuste de los parámetros de una red neuronal durante el entrenamiento (proceso de actualización) puede realizarse mediante múltiples algoritmos, siendo el más extendido la retropropagación de errores o *backpropagation*, un algoritmo, originalmente presentado para redes FFNN aunque posteriormente adaptado a otras variantes incluyendo las redes LSTM (*Long Short-Term Memory*, una variedad de red neuronal recurrente) empleadas en OpenNMT, la biblioteca utilizada en este proyecto.

En un entrenamiento por *backpropagation* el ajuste de los parámetros del modelo se realiza mediante la aproximación del gradiente, es decir, mediante la determinación de qué tipo de ajuste (cuáles ponderaciones aumentar, cuáles disminuir y en qué proporción) lleva a que el error (diferencia entre salida esperada y salida obtenida) decrezca en la mayor medida. Para cada unidad la salida obtenida o puede verse como una función de un vector de entrada $\overrightarrow{\mathbf{i}}$ (compuesto por las salidas de la capa anterior) y un vector con las ponderaciones $\overrightarrow{\mathbf{w}}$ (es decir, los parámetros a ajustar):

$$o = f(\overrightarrow{\mathbf{i}}, \overrightarrow{\mathbf{w}})$$

Tomando una instancia I a la vez (lo cual fija el vector $\overrightarrow{\mathbf{i}}$) se puede considerar que la salida es una función que depende únicamente de los parámetros a ajustar:

$$o_I = f_I(\overrightarrow{\mathbf{w}})$$

El concepto matemático de gradiente, notado como ∇, es una generalización de las derivadas para una función de múltiples variables (en este caso los parámetros, componentes del vector $\overrightarrow{\mathbf{w}}$) que refleja cómo se comporta una función ante una variación infinitesimal de de sus variables. El gradiente puede computarse como un vector compuesto por las derivadas parciales respecto a cada variable y corresponde a la dirección en la que la función muestra un mayor crecimiento. De esta forma, $\nabla f_I(\overrightarrow{\mathbf{w}})$ es un vector $\overrightarrow{v}$ tal que una variación infinitesimal $d\overrightarrow{v}$ en la dirección de $\overrightarrow{v}$ resulta en el mayor valor para $f_I(\overrightarrow{\mathbf{w}} + d\overrightarrow{v})$ y el menor valor para $f_I(\overrightarrow{\mathbf{w}} - d\overrightarrow{v})$.

Una vez determinada esta dirección, puede calcularse un nuevo conjunto de parámetros $\overrightarrow{\mathbf{w_1}} = \overrightarrow{\mathbf{w}} - \alpha\nabla f_I(\overrightarrow{\mathbf{w}})$ donde α o "paso" es un hiperparámetro escalar (potencialmente variable durante el entrenamiento) que puede interpretarse

como la magnitud del ajuste en una instancia. Para un valor de α lo suficientemente pequeño, se garantiza que:

$$f(\overrightarrow{\mathbf{i}}, \overrightarrow{\mathbf{w_1}}) < f(\overrightarrow{\mathbf{i}}, \overrightarrow{\mathbf{w}})$$

Esto equivale a que el nuevo conjunto de parámetros resulte en un menor error.

La iteración de este proceso se denomina "descenso por gradiente" y constituye una heurística para encontrar mínimos locales (vectores $\overrightarrow{\mathbf{w}}$ tales que cualquier variación pequeña resulta en un aumento del error total). Debe observarse que en un descenso por gradiente un paso α demasiado grande puede llevar a que $\overrightarrow{\mathbf{w}}$ sobrepase el mínimo local deseado y que no se tiene garantías de que el mínimo local hallado corresponda a un mínimo global.

En un algoritmo de *backpropagation*, el gradiente se calcula secuencialmente desde la última capa del modelo (*output layer*, para la cual la salida obtenida puede compararse directamente con la salida esperada para calcular el error) a las capas precedentes hasta llegar a la *input layer*. Esto resulta posible debido a que el efecto en cada parámetro de ponderación se limita a la salida de la *siguiente* capa. El gradiente global para los parámetros de la red neuronal se construye desde la capa final hacia la inicial, "propagando hacia atrás" el error obtenido, para luego poder realizar el ajuste de parámetros mediante descenso por gradiente. Esto permite realizar el cálculo de un modo eficiente.

Además de los algoritmos de *backpropagation*, resulta posible entrenar redes neuronales de diferentes características través de otros métodos como la neuroevolución (un enfoque que utiliza algoritmos evolutivos para explorar diferentes topologías y combinaciones de parámetros en una red neuronal) aunque los métodos basados en un descenso por gradiente como *backpropagation* continúan siendo la norma en la mayor parte de las aplicaciones del área.

Redes neuronales recurrentes (RNN)

Los primeros modelos de redes neuronales (perceptrones y redes neuronales prealimentadas o *feedforward neural networks*, *FFNN*) se caracterizaron por un flujo lineal de la información en el que la entrada pasa por una o más capas que progresivamente la transforman para generar una salida aunque sin que dichas capas mantengan un estado interno: la salida obtenida depende exclusivamente de la entrada actual, sin verse afectada por computaciones previas. Si bien esta característica puede resultar deseable en determinados escenarios (por ejemplo al permitir una más efectiva paralelización del procesamiento de múltiples elementos), resulta una limitación para los modelos de PLN al imposibilitar un procesamiento secuencia a secuencia en el que un elemento aporta contexto para la interpretación de los elementos subsiguientes.

En este tipo de modelos con un flujo lineal "no recurrente", el único modo en el que una entrada previa puede afectar las salidas futuras es mediante el ajuste de los parámetros del modelo (como los pesos o ponderaciones asociados a las conexiones entre unidades durante un entrenamiento), lo cual constituye

una “memoria a largo plazo” (*long-term memory*). Si bien este tipo de “memoria” resulta relevante para aplicaciones de PLN (por ejemplo al permitir definir modelos preentrenados), los ajustes graduales requeridos hacen infactible su uso para almacenar el contexto al procesar una secuencia en particular. (Hochreiter y Schmidhuber, 1997)

En oposición a estos primeros modelos, se han explorado múltiples arquitecturas de redes neuronales recurrentes (*recurrent neural network*, *RNN*) que mantienen un estado interno compuesto o influenciado por la propia salida de la red y capaz de persistir durante el procesamiento de una secuencia. Este estado interno, a veces referido como “memoria a corto plazo” (*short-term memory*) posibilita el tratamiento de una secuencia procesando un elemento a la vez pero manteniendo (“recordando”) el contexto que define la relación entre los elementos de la secuencia. Si bien este tipo de topologías más complejas que el simple flujo lineal de una *FFNN* permite tratar una mayor diversidad de problemas (entre ellos, la traducción de secuencias), su entrenamiento mediante el descenso por gradiente de error mediante *backpropagation* puede verse afectado por problemas como el desvanecimiento de gradiente (*vanishing gradient problem*): el efecto de un vector de error disminuye exponencialmente en cada iteración, rápidamente alcanzando un punto en el que el error asociado a las primeras entradas pierde toda influencia sobre la salida, lo cual restringe severamente el largo de las secuencias que la RNN puede procesar en forma efectiva. (Hochreiter y Schmidhuber, 1997)

La arquitectura LSTM (*Long Short-Term Memory*, “memoria a corto plazo larga”), propuesta en 1997 en el paper del mismo nombre (Hochreiter y Schmidhuber, 1997) ofrece una manera de mitigar este problema. Cada unidad en una red LSTM mantiene un vector de estado interno (o estado oculto) el cual influencia la salida generada y se retroalimenta de la misma. Si bien esto constituye un elemento común a otras arquitecturas de redes neuronales recurrentes, las redes LSTM se distinguen por regular tanto el efecto que el estado oculto tiene sobre la salida como viceversa a través de “puertas” (*gates*, típicamente funciones de activación sigmoides) controladas por parámetros que se pueden entrenar para reconocer en qué casos la información del contexto es relevante y en qué casos debe modificarse. En la implementación propuesta por Hochreiter, ilustrada en la figura 2.4, se cuenta con tres puertas: una “puerta de olvido” (*forget gate*) que determina a partir de la entrada y el estado oculto en qué grado el contexto contenido en el estado interno de la celda debe mantenerse u olvidarse, una “puerta de entrada” (*input gate*) que determina en qué grado la información de la entrada actual afectará al estado de la celda y una “puerta de salida” (*output gate*) que determina el estado oculto a utilizar en la siguiente iteración. El uso de unidades de puerta permite que el vector de error (diferencia entre valor esperado y valor obtenido, objetivo a minimizar por *backpropagation*) mantenga una escala constante en vez de decaer exponencialmente, lo cual permite mantener un contexto relevante para secuencias significativamente más largas que otras arquitecturas de RNN. (Hochreiter y Schmidhuber, 1997) (Singhal, 2020)

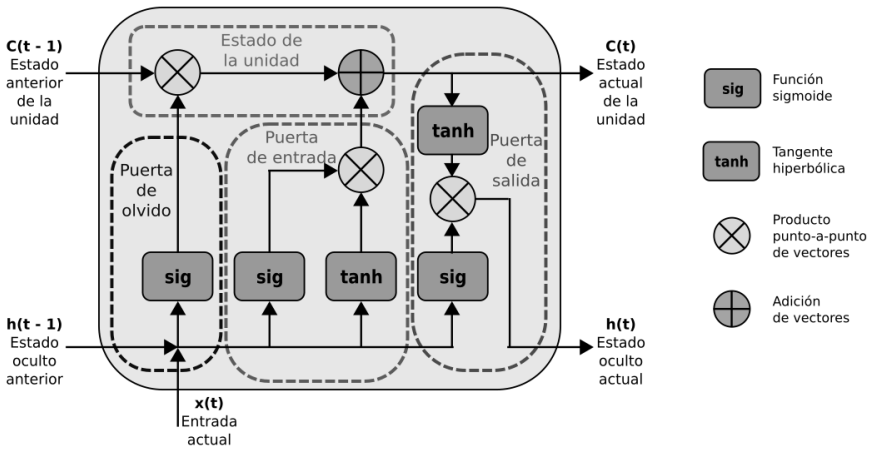

Figura 2.4: Esquema de una red LSTM. El estado de la unidad y el estado oculto generado como salida se computan en base a los estados anteriores y la entrada actual aplicando operaciones que determinan en qué medida el contexto existente afectará la nueva salida (puerta de olvido), de qué forma debe incorporarse la entrada actual al contexto (puerta de entrada) y cómo se computará la el siguiente estado oculto (puerta de salida). Diagrama tomado de Singhal (2020).

2.2.4. Modelos secuencia a secuencia

Los modelos secuencia a secuencia (*sequence to sequence*) constituyen un esquema común en las aplicaciones de procesamiento de lenguaje natural caracterizado por procesar una entrada como una lista secuencial de elementos (denominados *tokens*) a partir de la cual se genera una segunda secuencia de elementos, no necesariamente del mismo largo, a modo de salida. Un caso canónico de modelo secuencia a secuencia, ilustrado en la figura 2.5, se puede encontrar en la traducción automática entre dos lenguas naturales donde, por ejemplo, se parte de un texto en inglés y se busca producir un texto equivalente en español. En dicho escenario, resulta natural interpretar los textos de entrada y salida como secuencias de palabras en los idiomas respectivos aunque los *tokens* utilizados podrían incluir también marcas de puntuación u operar a nivel de morfemas, por ejemplo. Otros problemas clásicos de PLN que suelen tratarse como una transformación de secuencias son la generación de resúmenes automáticos, la transcripción de audio y la generación de respuestas a partir de una pregunta en lenguaje natural. (Dugar, 2019)

Actualmente el concepto de "secuencia a secuencia" (típicamente abreviado como *Seq2Seq*) suele hacer referencia a un enfoque introducido por investigadores de Google en el paper *Sequence to Sequence Learning with Neural Networks* (Sutskever, Vinyals, y Le, 2014) en el cual la secuencia de entrada se procesa a través de redes neuronales profundas (*Deep Neural Networks*, DNN), típicamente redes LSTM u otra forma de redes neuronales recurrentes (RNN),

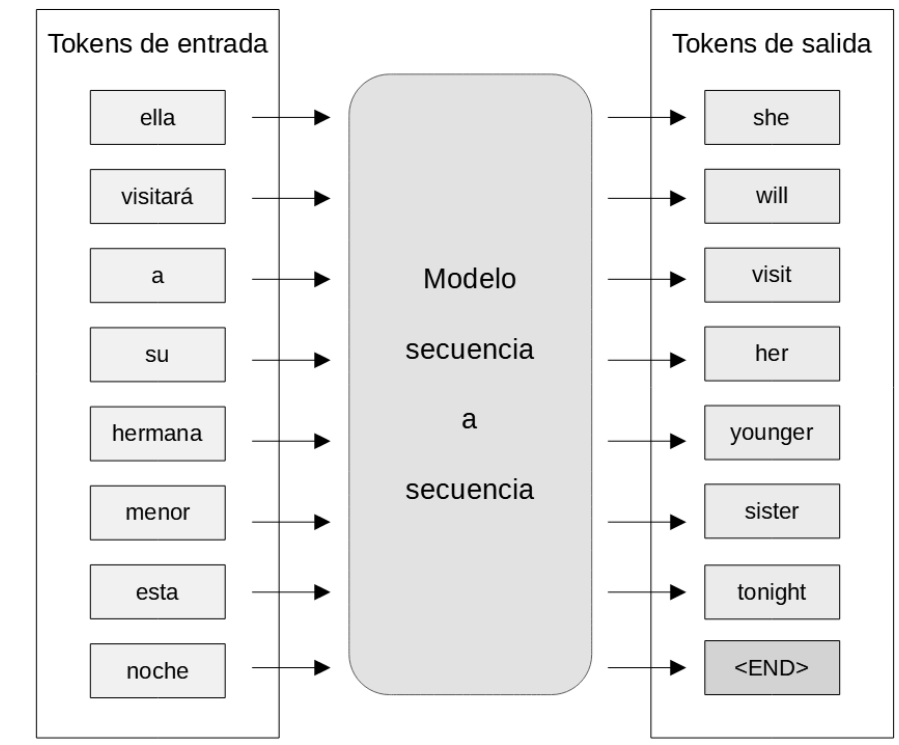

Figura 2.5: Esquema de un modelo secuencia a secuencia, ilustrado con una traducción automática de español a inglés. El modelo procesará secuencialmente elementos de entrada y generará elementos de salida. No necesariamente existirá una correspondencia uno-a-uno entre los elementos de la entrada y la salida. Usualmente los modelos secuencia a secuencia emitirán una salida especial (representada en el diagrama como *END*) para indicar el final de la secuencia generada.

estructuradas en dos componentes: un modelo denominado *encoder* que transforma la entrada (secuencia de largo variable) en una representación de largo fijo de la entrada y su contexto ("vector de estado oculto", *hidden state vector*), y un modelo denominado *decoder* que transforma dicha representación en la secuencia de salida (nuevamente de largo variable). En estos modelos el vector de estado oculto representa no solo un elemento específico dentro de la secuencia sino también su contexto. (Sutskever y cols., 2014) (Dugar, 2019)

En general un modelo *Seq2Seq* puede componerse de un pipeline con múltiples *encoders* y *decoders*, donde cada unidad produce su propio vector de estado oculto el cual se toma como input para las siguientes unidades. En los últimos años se ha explorado también el uso de modelos fundamentalmente compuestos de unidades de un solo tipo (únicamente *encoders* o *decoders*) que han obtenido resultados de estado del arte en múltiples ámbitos del Procesamiento de Lenguaje Natural como es el caso de BERT (Devlin, Chang, Lee, y Toutanova, 2018), un modelo basado en *encoders* capaz de transformar texto en lenguaje natural en una representación que facilita su procesamiento efectivo en un gran número de tareas, y GPT-3, un modelo basado en *decoders* capaz de generar texto similar al redactado por un ser humano. (Dugar, 2019) (Yang, 2020)

Las implementaciones actuales de modelos *Seq2Seq* típicamente incorporan mecanismos de atención, una técnica que permite ponderar los valores sobre los cuales operará una red neuronal (como los elementos de la secuencia de entrada y el vector de estado oculto producido tras procesar el último paso) detectando relaciones entre los elementos de una o más secuencias. Estos mecanismos subsanan una deficiencia encontrada en los primeros modelos basados en redes neuronales recurrentes: la información contenida en los primeros elementos de la secuencia de entrada tiende a perder relevancia en los pasos sucesivos, lo cual resulta en una efectividad disminuida para secuencias largas; la incorporación de un mecanismo de atención permite poder destacar los valores derivados de estos primeros elementos en los puntos de la secuencia de salida para los que resultan más relevantes, permitiendo por lo tanto aprovechar mejor la representación de contexto y aumentando el rendimiento para secuencias largas. El uso de mecanismos de atención se incorpora a los modelos de *encoders* y *decoders* en la arquitectura de transformadores o *transformer*, introducida en el paper *"Attention Is All You Need"* (Vaswani y cols., 2017), obteniendo resultados *state of the art* en problemas de traducción automática. (Yang, 2020)

Actualmente se dispone de múltiples frameworks que facilitan la generación de modelos secuencia para tareas relacionadas al procesamiento de lenguaje natural. En este proyecto se utilizó la plataforma OpenNMT (Klein, Kim, Deng, Senellart, y Rush, 2017), orientada a la traducción automática, empleando un modelo *encoder-decoder* basados en redes LSTM e incorporando un mecanismo de atención.

2.2.5. Tokenización y embeddings

Las tareas de procesamiento de lenguaje natural y, en particular, el uso de enfoques secuencia a secuencia, requieren transformar la entrada en una serie de

elementos discretos denominados *"tokens"* mientras que la salida se compondrá asimismo de una secuencia de tokens. En el caso de un problema de procesamiento de lenguaje natural, los tokens típicamente corresponden a palabras (por ejemplo al segmentar el string de entrada por los caracteres de espacio) o a caracteres de puntuación aunque, dependiendo del problema a tratar, se podría incluir otros elementos como marcas especiales (por ejemplo separadores en modelos que toman como entrada una tupla de oraciones), representaciones de morfemas, caracteres individuales, fragmentos de palabras, etc. Independientemente de la naturaleza de los tokens utilizados se utiliza el término "vocabulario" para denominar al conjunto de tokens considerados por el modelo (por ejemplo, las palabras y marcas de puntuación presentes en un corpus utilizado durante el entrenamiento del modelo). El *vocabulario* correspondiente a la entrada puede o no coincidir con el vocabulario de la salida del modelo: en el caso de un modelo de traducción automática ambos vocabularios corresponderán a idiomas diferentes. (OpenNMT, 2020)

En el procesamiento de un texto en lenguaje natural típicamente existe la posibilidad de que la entrada contenga una palabra no contemplada dentro del vocabulario (por ejemplo un nombre propio). El problema de los elementos "fuera de vocabulario" (*out of vocabulary*, OOV) admite varios tratamientos; OpenNMT opta por un enfoque clásico consistente en emplear un token especial (representado con el símbolo $<unk>$) que es utilizado como reemplazo para cualquier elemento OOV. Una alternativa consiste en utilizar *"wordpieces"*: un conjunto de substrings cuya combinación permite representar cualquier secuencia de caracteres, aunque al costo de que un mismo elemento léxico pasará a estar representado por múltiples tokens. (OpenNMT, 2020)

En el caso del presente trabajo, el vocabulario de entrada sigue el criterio de tokenización empleado en el corpus AnCora donde los tokens corresponden a palabras individuales, marcas de puntuación o grupos léxicos (particularmente para nombres propios, incluso si los mismos podrían analizarse como una frase en si misma: el nombre de la "Real Academia de la Historia", una institución académica madrileña, se representa como un solo token *real_academia_de_la_historia* en vez de con los tokens de las cinco palabras que lo componen). Se sigue el criterio de OpenNMT de tratar términos OOV con un único token especial. Como se explicará más adelante, para la salida se optó por utilizar un vocabulario muchísimo más reducido basado principalmente en clases de palabras (*part of speech*, POS) con el objetivo de disminuir los requisitos de memoria requeridos para el procesamiento.

Dado que las redes neuronales operan sobre valores numéricos (típicamente vectores), resulta necesario transformar los tokens en una representación numérica conocida como *embedding*. Si bien resulta posible obtener resultados válidos asignando a cada token un vector con valores aleatorios (como los *embeddings* generados por defecto en OpenNMT), a menudo se obtiene resultados mejores al utilizar *embeddings* donde los componentes del vector asignado a una palabra dada guardan una determinada relación con los de los vectores asociados a las palabras que suelen aparecer en su mismo contexto. En efecto, esto lleva a que el vocabulario quede "embebido" en un espacio con la dimensionalidad del

vector utilizado para el *embedding* (de 300 elementos en el caso de este estudio) donde la distancia entre vectores corresponde a una similitud en los contextos asociados a cada palabra los cuales suelen ser representativos de su semántica. Las bibliotecas de procesamiento de lenguaje natural como OpenNMT suelen admitir colecciones de *embeddings* "preentrenados" (en tanto los valores asignados a cada token se derivan del procesamiento de grandes volúmenes de texto en el idioma correspondiente), los cuales pueden aplicarse en múltiples proyectos. (OpenNMT, 2020)

2.2.6. Embeddings preentrenados

Existen múltiples técnicas para generar *embeddings* de palabras representativos de sus contextos, siendo las dos variantes más extendidas la técnica *Word2vec* (Mikolov, Chen, Corrado, y Dean, 2013) y la técnica *Global Vectors* o *GloVe* (Pennington, Socher, y Manning, 2014b). Ambos enfoques parten de un corpus extenso (en el orden de cientos o miles de millones de palabras) y utilizan técnicas de aprendizaje automático no-supervisado para asignar representaciones vectoriales en los que la similitud entre vectores (medida utilizando una función de métrica como la distancia euclídea o la similitud coseno) corresponde a una similitud semántica. Esto último comúnmente se ilustra por la posibilidad de reconocer relaciones análogas a través de la adición y substracción de vectores: si se tiene la correspondencia entre "hombre" y "mujer", se puede obtener la relación análoga entre "rey" y "reina" observando que el *embedding* que mejor aproxima a $embedding(rey) + (embedding(mujer) - embedding(hombre))$ es el *embedding* de "reina". De la misma manera se pueden extraer analogías entre países y capitales, formas verbales en presente y en pasado o entre adjetivos y superlativos. Más allá de las potenciales aplicaciones directas de esta característica emergente de los modelos *Word2vec* y *Glove* (por ejemplo sugerir sinónimos aplicables en un contexto dado), el hecho de que la similitud semántica tiende a correlacionarse con una similitud sintáctica permite que incorporar este tipo de *embeddings* a aplicaciones de PLN como los problemas de traducción secuencia-a-secuencia resulten en una notoria mejora en la performance y calidad de los resultados. (Mikolov y cols., 2013)

La creación de colecciones de *embeddings* preentrenados se realiza mediante aprendizaje automático no supervisado, detectando una estructura a partir de un modelo no etiquetado. (Delua, 2021)

Los modelos *Word2vec* utilizan una red neuronal poco profunda (típicamente de dos capas) que intentan predecir la ocurrencia de una palabra dada en base a su contexto local mientras que los modelos *GloVe* parten de una matriz de coocurrencia de palabras creada a partir de todo el corpus sobre la cual se emplean técnicas de factorización para obtener un conjunto de vectores cuya correlación permite reconstruir la matriz de coocurrencia con un mínimo nivel de error. (Pennington, Socher, y Manning, 2014a)

Las colecciones de *embeddings* generadas por ambas técnicas dependen primordialmente de dos parámetros: la dimensionalidad de los vectores a construir (donde una mayor cantidad de componentes por vector permite reconstruir in-

terrelaciones más ricas aunque al costo de una mayor complejidad computacional) y el corpus utilizado (el cual preferentemente habrá de comprender una gran variedad de textos para minimizar posibles sesgos). Las primeras colecciones de embeddings presentadas se construyeron a partir de múltiples corpus en idioma inglés, como un corpus de seis mil millones de de palabras obtenido de Google News en (Mikolov y cols., 2013).

Actualmente se dispone de colecciones de *embeddings* preentrenados en base a corpus en español entre los que se incluye una colección generada por aplicando la técnica *Word2Vec* sobre un corpus de dos mil millones de palabras (Azzinari y Martínez, 2016) y las múltiples colecciones generadas por el Departamento de Ciencias de la Computación de la Universidad de Chile en base al *Spanish Billion Word Corpus* (Cardellino, 2019) aplicando múltiples técnicas, incluyendo *Word2vec* y *GloVe*. (Rojas, Cañete, y Nguyen, 2018)

2.2.7. Corpus de entrenamiento

La creación de un modelo de *Machine Learning* requiere del uso de un conjunto amplio de ejemplos a partir de los cuales el modelo será capaz de reconocer patrones relevantes para el problema abordado que sean también aplicables a instancias no incluidas dentro del conjunto de ejemplos original; en otras palabras, el "aprendizaje" debe *generalizar*. En el contexto del aprendizaje automático aplicado al procesamiento del lenguaje natural se emplean colecciones de textos referidas como "corpus".

Dependiendo del problema a resolver y la disponibilidad de datos se puede realizar un aprendizaje automático supervisado en el cual se busca que el modelo aprenda a resolver una tarea específica en base a ejemplos preetiquetados que relacionan entradas con la salida esperada o, por el contrario, se puede realizar un aprendizaje automático no supervisado donde se presenta a un modelo un gran volumen de datos no preetiquetados y se espera que el modelo en sí pueda identificar una estructura interna, por ejemplo detectando agrupamientos de elementos según una determinada métrica de similaridad (*clustering*), como ocurre en la construcción de word-embeddings preentrenados mencionada en la sección previa. (Delua, 2021)

En el caso de modelos secuencia-a-secuencia como el propuesto en este trabajo resulta natural adoptar un enfoque de aprendizaje automático supervisado en el que se le suministra al modelo una entrada (habitualmente el texto "a traducir") aparejado de la salida esperada (la "traducción"). Dado que en el presente trabajo se busca identificar los casos en los que una oración presenta un sujeto tácito u omitido, resulta necesario emplear una colección de texto para la cual se haya etiquetado esta característica, como es el caso con el corpus AnCora, empleado en este proyecto.

2.2.8. Métricas de evaluación

La detección de sujetos omitidos puede verse como un problema de clasificación, ya sea entre dos clases (oraciones con y sin un sujeto explícito) o entre

tres (sujetos explícitos, sujetos omitidos y oraciones impersonales). Para la evaluación de este tipo de problemas existe un conjunto de métricas establecidas en la literatura entre las que se destaca la exactitud o *accuracy*, la precisión, la exhaustividad o *recall* y la medida F1 (también referida en la literatura como valor F1 o valor F).

Para el cómputo de estas métricas se debe considerar el resultado obtenido y el resultado esperado para cada instancia dentro del conjunto sobre el que se evalúa (corpus de evaluación). Estos datos pueden representarse mediante una **matriz de confusión** en la cual se indica cuántas veces se obtuvo cada resultado para entradas de cada categoría real. Si el problema de clasificación maneja n categorías, la matriz de confusión tendrá dimensiones $n \times n$ donde el elemento a_{ij} indicará cuántas instancias se clasificaron con la categoría i siendo el resultado correcto la categoría j. De esta manera, un clasificador sin errores se caracterizaría por una matriz de confusión diagonal, donde a_{ii} incluye a todas las instancias del corpus de evaluación en la categoría i mientras que $a_{ij} = 0$ para todo $i \neq j$.

Para una categoría dada, la clasificación de cada instancia puede corresponder a uno de cuatro escenarios:

- **Verdadero Positivo (VP)**: la instancia pertenece a la categoría y se la clasifica correctamente.
- **Falso Positivo (FP)**: la instancia no pertenece a la categoría, pero se la identifica incorrectamente.
- **Verdadero Negativo (VN)**: la instancia no pertenece a la categoría, y no se la reconoce como tal.
- **Falso Negativo (FN)**: la instancia pertenece a la categoría, pero no se la identifica correctamente.

Estos valores se utilizarán para las métricas que se presentarán a continuación.

Accuracy

La exactitud o *accuracy* corresponde al porcentaje de clasificaciones correctas. Esta medida puede calcularse en general, contemplando todas las categorías manejadas por el clasificador.

$$ACC = \frac{correcto}{total}$$

En el caso de una clasificación binaria o para calcular el *accuracy* al considerar solamente una categoría, la métrica se puede calcular como la proporción que representan los verdaderos positivos y los verdaderos negativos sobre el total:

$$ACC = \frac{VP + VN}{VP + FP + VN + FN}$$

Si bien esta medida resulta muy intuitiva, puede resultar un tanto inadecuada cuando la distribución de las categorías es desigual. A modo de ejemplo, si los ejemplos positivos representaran el 99 % de las instancias a evaluar, un clasificador A que siempre devolviera resultado positivo siempre obtendría un *accuracy* de 99 % pese a que garantiza que el 1 % de instancias negativas resultará en un falso positivo. Si se lo comprara con un clasificador B que clasifica correctamente el 90 % de las entradas positivas y el 90 % de las entradas positivas se obtendría un *accuracy* inferior, de un 90 % pese a que en la práctica la capacidad de encontrar verdaderos negativos del clasificador B podrían hacerlo preferible.

$$ACC_A = \frac{99 \times 1 + 1 \times 0}{100} = 0{,}99 > ACC_B = \frac{99 \times 0{,}90 + 1 \times 0{,}90}{100} = 0{,}90$$

Este efecto debe tenerse en cuenta en el problema de detección de sujeto omitido ya la distribución de oraciones con sujeto explícito, con omitido o impersonales es considerable desbalanceada; en el corpus AnCora se observa que cerca de un 75 % de las oraciones presentan un sujeto explícito.

Precision y *recall*

La precisión o *precision* y el *recall* o exhaustividad son dos medidas complementarias que permiten analizar la calidad de los resultados obtenidos para una categoría en particular, independientemente de si la distribución de las instancias es poco balanceada.

La precisión evalúa el grado de confianza que se le puede dar a una clasificación positiva, comparando el ratio entre clasificaciones positivas correctas (verdaderos positivos) y el total de instancias *clasificadas como positivas* (verdaderos positivos y falsos positivos).

$$PRE = \frac{VP}{VP + FP}$$

Esto supone que un sistema con una precisión del 100 % solamente devolverá resultados positivos correctos, aunque esto no asegura que no haya instancias positivas que no se hayan reconocido (falsos negativos). A modo de caso extremo, un sistema que etiquete como positiva solo 1 entrada en 100 positivas pero que no obtenga falsos positivos presentará una precisión de 100 %. En caso que un clasificador no reconozca ninguna instancia positiva la métrica queda aritméticamente indefinida ya que el denominador $VP + FP = 0$.

Por el contrario, la exhaustividad o *recall* mide qué proporción de las instancias *realmente positivas* se reconocen, es decir, el número de clasificaciones positivas correctas (verdaderos positivos) frente al total de instancias positivas, correctamente clasificadas o no (verdaderos positivos y falsos negativos).

$$RCL = \frac{VP}{VP + FN}$$

El *recall* presenta un uso y problemas complementarios a los de la precisión: permite reconocer la incidencia de los falsos negativos (casos positivos que no se pudieron reconocer) pero no tiene en cuenta el efecto de los falsos positivos. Un clasificador que devolviera siempre un resultado positivo mostraría un *recall* del 100 % al ser capaz de recuperar todas las instancias positivas aunque clasificando erróneamente todas las instancias negativas. Esta complementaridad lleva a que ambas métricas se suelan emplear en conjunto.

Para un problema con múltiples categorías (como la clasificación tripartita entre sujeto omitido vs explícito vs impersonal) se tendrá un valor de precisión y *recall* para cada categoría, tomando como "positivo" a los valores de la categoría considerada y como "negativo" a los restantes. Esto implica que, por ejemplo, de tomar a "sujeto omitido" como categoría, una clasificación de un verbo impersonal como explícito habría de considerarse como un "verdadero negativo" (no es un sujeto omitido ni se lo clasifica como tal) pese a que las categorías esperada y obtenida no coinciden. Esto no resulta problemático dado que los verdaderos negativos VN no participan del cálculo de estas dos métricas.

Medida F1

La medida F1 (también llamada medida F, valor F1 o valor F; *F1 score* en inglés) constituye una forma de combinar las métricas de precisión y *recall* para una categoría por lo que resulta representativa tanto de la habilidad del clasificador para reconocer las instancias positivas (medido por el *recall*) como del grado de confianza que se le puede dar a las clasificaciones positivas (medido por la precisión). Por esta razón la medida F1 suele utilizarse como la principal métrica a comparar entre sistemas de clasificación.

Esta métrica se calcula como la media armónica de ambos valores:

$$F1 = \frac{2 \times PRE \times RCL}{PRE + RCL} = \frac{VP}{VP + \frac{1}{2}(FP + VN)}$$

Debido a su método de cálculo, la medida F1 asigna la misma importancia a los errores resultantes de un **FP** o de un **FN**. Debe notarse que en algunos contextos los errores derivados de falsos positivos y de falsos negativos (referidos también como errores de tipo I y tipo II, respectivamente) pueden tener diferentes niveles de severidad en la práctica. A modo de ejemplo, si un clasificador intentara detectar la toxicidad de un producto resultaría mucho más severo obtener un falso negativo (no reconocer un artículo nocivo, pudiendo resultar en una intoxicación) que un falso positivo (que llevaría simplemente a descartar innecesariamente un artículo apto para consumirse). En estos casos, resultará más importante priorizar otras métricas como el *recall*.

Medida macro F1 y medida macro F1 ponderada

Dado que la medida F1 se computa a partir de la precisión y el *recall*, comparte con estas métricas la desventaja de solamente aplicar para una categoría específica. Para un problema de clasificación tripartita como el propuesto se

tendrá tres medidas F1: una para el reconocimiento de oraciones con sujeto omitido, otra para el de oraciones con sujeto explícito y una tercera para oraciones impersonales.

Existen maneras de agregar los datos entre categorías, siendo las más extendidas la medida macro F1 en su variante simple y ponderada. La medida macro F1 consiste simplemente en el promedio de las medidas F1 para cada categoría:

$$F1_{Macro} = \frac{\sum F1_i}{n}$$

Ante conjuntos de datos con una distribución de categorías poco balanceada, puede resultar más apropiado aplicar un promedio ponderado en el que el peso que se le dará a cada medida F1 dependerá del número de instancias positivas para la categoría correspondiente (conocido como el *soporte* de la categoría):

$$F1_{Ponderado} = \frac{\sum(F1_i \times Soporte_i)}{n \times \sum Soporte_i}$$

Métricas para evaluar entrenamiento

El procesamiento de una entrada en este trabajo se basa en la salida de un modelo secuencia a secuencia de OpenNMT, a partir de la cual se buscará extraer las clasificaciones correspondientes a cada verbo finito. El entrenamiento de dichos modelos se realiza mediante un aprendizaje supervisado utilizando secuencias de entrada y salida generadas a partir del corpus AnCora, las cuales incluyen elementos para la detección de sujetos omitidos y cláusulas impersonales aunque cabe reconocer una distinción entre la tarea de traducción entre secuencias para la que se entrena el modelo en OpenNMT y el problema de clasificación en sí. Debido a esto resulta de utilidad contar con métricas que permitan evaluar cómo evoluciona el comportamiento del modelo Seq2Seq en sí durante su entrenamiento, analizando la calidad de las "traducciones" entre secuencias que genera lo cual permitirá determinar en qué momento el modelo comienza a converger, es decir, a alcanzar un óptimo local que no mejorará con rondas adicionales de backpropagation.

Se proponen dos métricas para evaluar el rendimiento del modelo secuencia-a-secuencia: el porcentaje de coincidencias exactas y la tasa de error en traducciones (generalmente referida como TER, acrónimo del inglés *Translation Error Rate*).

Se considera que una entrada genera una coincidencia exacta si la salida obtenidas del modelo Seq2Seq coincide con la salida esperada, presentando todos los mismos tokens en el orden correcto. El porcentaje de coincidencias exactas de un modelo se define como el ratio entre las entradas del corpus sobre el que se evalúa para las que se genera coincidencias exactas. Debe observarse que esta métrica presenta una dependencia con el largo de las salidas esperadas: cuanto mayor sea la cantidad de tokenes en una secuencia de salida mayor será la probabilidad de que el modelo cometa algún error, por lo que el porcentaje de

coincidencias exactas será superior para corpus compuestos de secuencias cortas e inferior para corpus que presenten secuencias largas.

La tasa de error en traducciones, o TER, mide la calidad de una traducción entre secuencias cuantificando el número de correcciones que hace falta efectuarle a la salida obtenida para que coincida con la salida esperada. Para esto se calcula la distancia de Levenshtein entre ambas secuencias (el menor número de operaciones de inserción, eliminación o sustitución de tokens individuales requerido para transformar una secuencia a otra) y se la divide por la cantidad de tokens en la secuencia esperada (de forma que la métrica no presente un sesgo frente a secuencias cortas como en la coincidencia exacta). A modo de ejemplo, si la secuencia esperada fuera "1 2 3 4 5 6" y la secuencia obtenida fuera "1 7 2 3 8 5' se puede computar que la distancia de Levenshtein entre ambas es de 3 (la secuencia obtenida se puede transformar en la secuencia esperada mediante tres operaciones: la eliminación del token "7", la sustitución del token "8" por "4" y la inserción del token "6") por lo que la tasa de error en el ejemplo es de $\frac{3}{6} = 0{,}5$ o de un 50 %. La TER de un modelo dado puede obtenerse como el promedio de la TER para cada instancia del corpus empleado para la evaluación.

El objetivo del problema de optimización manejado por OpenNMT es que la traducción entre secuencias aproxime las salidas esperadas en el corpus de entrenamiento por lo que en el caso ideal un modelo presentaría un porcentaje de coincidencias exactas del 100 % y una TER de 0. Debe observarse, sin embargo, que la salida de estos modelos es en realidad un elemento auxiliar para el problema planteado en este proyecto; no resulta necesario que las salidas obtenidas coincidan con las salidas esperadas para que su procesamiento permita clasificar correctamente las oraciones contenidas.

Capítulo 3

Revisión de antecedentes

Dado que este estudio se enfoca en la aplicación de una técnica (la aplicación de modelos de traducción automática a otras tareas de PLN) al problema del reconocimiento de sujeto omitido en español, cabe considerar los antecedentes tanto respecto a la técnica como al problema en sí.

3.1. Aplicación de técnicas de traducción automática a problemas de *parsing*

En *Grammar as a Foreign Language* (Vinyals y cols., 2014) se abordó el problema del análisis de constituyentes sintácticos o *parsing* a través de un modelo secuencia-a-secuencia con mecanismos de atención concebido originalmente para tareas de traducción automática. Este nuevo enfoque presentó múltiples ventajas frente a los modelos construidos específicamente para resolver un problema de parsing: al utilizar un modelo secuencia-a-secuencia más genérico se obtiene una mayor flexibilidad que permite adaptar el enfoque a otras tareas, detección de otros modelos sintácticos o el análisis de idiomas con otras características sintácticas, los modelos resultantes operan con mayor rapidez que los parsers tradicionales (cuyo tiempo de procesamiento escala en forma cúbica respecto al número de tokens a procesar, frente a un tiempo lineal en modelos Seq2Seq simples o cuadrático en modelos Seq2Seq con mecanismos de atención) y obteniendo una performance comparable a los mismos al entrenar con un corpus pequeño (cuarenta mil oraciones) etiquetado manualmente y resultados de estado del arte al emplear un corpus considerablemente mayor (once millones de oraciones) etiquetado mediante parsers tradicionales. (Vinyals y cols., 2014)

Dado que la salida esperada en un análisis de constituyentes sintácticos tiene estructura de árbol, se requiere un algoritmo reversible de linealización que permita transformar los árboles sintácticos de los corpus de entrenamiento en secuencias lineales de tokens aptas para el modelo Seq2Seq, así como para reconvertir la salida del mismo al formato de árbol sintáctico. En el caso del paper citado, esto puede resolverse mediante un simple parentizado, aunque la técni-

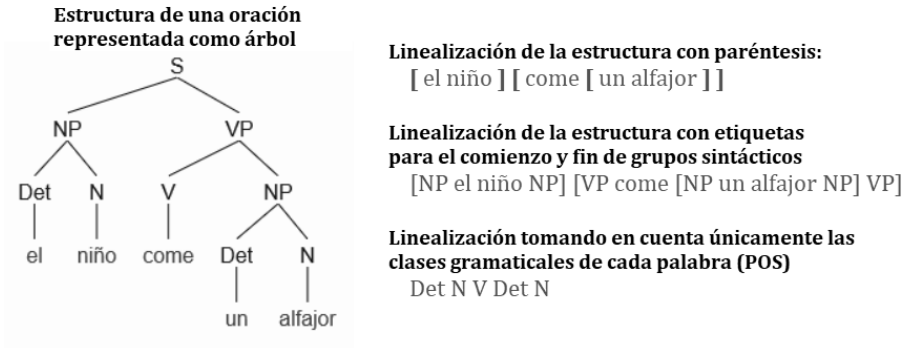

Figura 3.1: Ejemplos de cómo un árbol sintáctico puede transformarse en una secuencia lineal de tokens. El primer ejemplo emplea solo dos tokens especiales (correspondientes a paréntesis) que permiten reconstruir la estructura arbórea original. El segundo ejemplo incorpora distintos tokens para delimitar grupos sintácticos nominales y verbales. El tercer ejemplo descarta la información estructural pero preserva información sobre las categorías léxicas de cada palabra (*POS-tagging*).

ca podría aplicarse a problemas cuya salida esperada tenga una estructura más compleja que requiera un proceso de linealización más complejo. En la figura 3.1 se ilustran algunos algoritmos posibles para este tipo de problemas (Vinyals y cols., 2014)

En Vinyals y Kaiser (2015) se observa al aplicar la técnica empleando un modelo con redes LSTM sin mecanismos de atención los resultados del modelo de deterioraban considerablemente al aumentar el largo de la secuencia a procesar, presentando una medida F1 superior a 0.93 para oraciones de menos de 30 tokens, la cual cae por debajo de 0.91 para oraciones de 60 tokens o más; un resultado coincidente al obtenido con el parser de Berkeley (*BerkeleyParser*), un parser tradicional de complijidad $O(n^3)$ entrenado sobre el mismo corpus tomado como baseline para el trabajo. Por el contrario, al emplear un modelo Seq2Seq con redes LSTM y mecanismos de atención (LSTM+A) se obtuvieron resultados considerablemente superiores a los de la baseline (con una medida F1 por encima de 0.93) y con una menor tasa de degradación en relación al largo de la secuencia de entrada. (Vinyals y cols., 2014)

En Stanovsky y Dagan (2018) se aplicó este mismo enfoque a un problema distinto, el análisis de dependencias semánticas, publicando el paper *Semantics as a Foreign Language*. El problema tratado en dicho trabajo presenta características más complejas: mientras que un *parsing* de constituyentes sintácticos resulta en un árbol con raíz muy sencillo de linealizar, la representación de la estructura semántica de un enunciado suele adoptar formas más complejas como un grafo o un multigrafo, cuya linealización no resulta trivial. Stanovsky y Dagan consideran múltiples representaciones semánticas (cada una con un esquema

de linealización) las cuales se entrenan en un mismo modelo Seq2Seq utilizando técnicas análogas al entrenamiento de un modelo de traducción multilingüe (donde se emplean etiquetas especiales para indicar el "idioma" de la entrada y la salida; en este caso, múltiples representaciones semánticas linealizadas). Este trabajo también obtuvo un rendimiento similar al de los sistemas de estado del arte para el problema planteado, aunque con una complejidad computacional menor. (Stanovsky y Dagan, 2018)

3.2. Antecedentes en reconocimiento de sujeto omitido

En la literatura se encuentran múltiples ejemplos de trabajos orientados a la detección automática de sujetos implícitos y construcciones impersonales en español. Los primeros avances en este aspecto incluyen a *A Computational Approach to Zero-pronouns in Spanish* (Ferrández y Peral, 2000), donde la detección de sujetos omitidos se realiza a través de una serie de reglas basadas en heurísticas para intentar reconocer un posible sujeto explícito entre los argumentos presentes en la oración. En dicho estudio se distinguió únicamente entre sujetos explícitos e implícitos, sin considerarse por separado las construcciones impersonales. El estudio presentó una *accuracy* general de 88 %, alcanzando una medida F1 de 0,927 para el reconocimiento de sujetos omitidos en el corpus *LexEsp* (basado en textos periodísticos) y de 0,783 sobre el parte del contenido de *The Blue Book* o "Libro Azul" (un manual sobre políticas para autoridades de telecomunicaciones). Si bien los resultados obtenidos por Ferrández presentan un rendimiento muy elevado (la medida F1 de 0,927 para sujetos omitidos es el valor más alto encontrado en la literatura), estudios posteriores como (Rello y cols., 2012) señalan como problemática la no falta de consideración de los verbos impersonales como un fenómeno distinto a la existencia de un sujeto omitido (o un "pronombre nulo" en términos de Ferrández y Peral) así como el uso de corpus muy limitados (906 verbos para LexEsp y 693 para The Blue Book) con una distribución de ejemplos que difiere de la observada en los datos empleados por los demás estudios (LexEsp presenta un 46 % de verbos con sujeto omitido, frente a un 25 % en corpus como AnCora o Elliphant), por lo que Rello (Rello y cols., 2012) considera que los resultados de este primer estudio no resultan comparables a los siguientes.

En base a experiencias previas que demostraron un mejor rendimiento en la identificación de construcciones no-referenciales en inglés y francés al aplicar técnicas de aprendizaje automático en vez de reglas heurísticas, el problema del reconocimiento de sujeto omitido en español se abordó a través de aprendizaje automático por primera vez en *Elliphant: Improved Automatic Detection of Zero Subjects and Impersonal Constructions in Spanish* (Rello y cols., 2012). Para este trabajo, los autores compilaron un corpus basado en textos legales y artículos médicos (ESZIC, parte del corpus posteriormente publicado como Elliphant); en este caso realizando una distinción tripartita entre oraciones con

sujeto explícito, oraciones con sujeto omitido y oraciones impersonales. Rello y sus colaboradores no emplearon un modelo basado en redes neuronales sino un clasificador de clustering con el algoritmo $K*$ (Cleary y Trigg, 1995), el cual buscaba predecir la categoría correcta para un verbo finito a partir de 14 atributos entre las que se incluía la clasificación mediante un parser basado en reglas (lo cual ofrecía un baseline de 75 % de *accuracy* para detectar sujetos explícitos), el lexema del verbo, los cuatro tokens anteriores y posteriores, la presencia del pronombre "se" antepuesto al verbo, entre otros. La performance de este clasificador sobre el corpus ESZIC presentó un *accuracy* de 87,3 % para sujetos explícitos, 87,4 % para omitidos y 98,8 % para verbos impersonales, con medidas F1 de 0,912, 0,755 y 0,727 respectivamente. (Rello y cols., 2012)

En 2018, González y Martínez retomaron y desarrollaron el trabajo de Rello y colaboradores en *Detección de sujetos omitidos en el español*. En dicho proyecto, las autoras pasaron a utilizar el corpus AnCora, considerado más representativo que ESZIC para textos en general al presentar un dominio menos específico (artículos periodísticos, frente a los textos legales y artículos médicos manejados en Elliphant), aunque los modelos propuestos se evaluaron frente a ambos corpus con el fin de poder comparar los resultados de ambos estudios. González y Martínez incorporaron nuevos atributos a los propuestos por Rello y colaboradores, los cuales se emplearon para el entrenamiento de múltiples modelos de clasificación, incluyendo regresión logística, SVM (*Support Vector Machines*) y diferentes variantes del clasificador bayesiano ingenuo (*Naïve Bayes*) así como métodos que combinan los resultados de otros clasificadores como el clasificador por votación o *Voting Classifier*, el cual presentó los mejores resultados del estudio con un *accuracy* general de 83,3 % y medidas F1 de 0,909 para reconocimiento de sujetos explícitos, 0,718 para omitidos y 0,555 para impersonales respecto al corpus Ancora y de 0,891, 0,663 y 0,622 respecto al corpus ESZIC de Elliphant. (González y Martínez, 2018)

El trabajo de González y Martínez fue extendido por Gerez, Irazusta e Irace en *Detección y resolución de sujetos nulos*, presentado en 2019. Este estudio incorporó el uso de clasificadores basados en redes neuronales, incluyendo redes prealimentadas simples (FFNN, *Feed-Forward Neural Network*), redes convolucionales (CNN) y redes neuronales recurrentes LSTM aunque sin mecanismos de atención. Se identificó que este tipo de clasificadores obtenía una performance superior a la de los trabajos previos, particularmente con un modelo con redes convolucionales que alcanzó un *accuracy* de 82 % y medidas F1 de 0,89, 0,67 y 0,34 para un corpus que combina los textos médicos y legales de ESZIC y los textos periodísticos de AnCora. Si bien este rendimiento superó al del clasificador presentado por González y Martínez respecto a la distinción binaria entre sujetos explícitos y sujetos omitidos, se obtuvo un rendimiento menor en el reconocimiento de cláusulas impersonales. Los autores observaron que para un modelo de SVC la detección de cláusulas impersonales mejoraba si se aumentaba artificialmente el número de ejemplos de verbos finitos impersonales en el corpus de entrenamiento mediante la repetición de ejemplos de dicha categoría. (Gerez y cols., 2019)

En los estudios citados se aprecia una gran diversidad de enfoques para el

tratamiento del problema aunque utilizando diferentes corpus para su entrenamiento y evaluación. Los trabajos centrados en aprendizaje automático coinciden en aplicar un esquema de categorización tripartita, clasificando las cláusulas verbales (dominadas por un verbo finito) en instancias con sujeto explícito, sujeto implícito o de verbo impersonal. Los resultados apuntan consistentemente a un mejor rendimiento en los sistemas basados en aprendizaje automático frente a los basados en otras heurísticas, con el trabajo de Gerez y colaboradores presentando los mejores resultados al emplear redes neuronales de topologías recurrentes. Se observa que ninguno de los trabajos encontrados en la literatura para la clasificación del sujeto omitido en español incorpora mecanismos de atención como los empleados en OpenNMT, la herramienta utilizada en este proyecto.

Capítulo 4

Desarrollo

El objetivo de este trabajo es probar la viabilidad de utilizar un modelo de traducción automática basado en OpenNMT para la detección de una característica gramatical: la presencia o ausencia de sujeto omitido. Para esto, deben definirse los siguientes elementos:

- El corpus de datos a utilizar para el entrenamiento.
- Los formatos de entrada y de salida a utilizar en el modelo.
- El modo en el que se identificará la categoría de la oración a partir de la salida de OpenNMT.
- Uso de word-embeddings preentrenados.
- Hiperparámetros del modelo incluyendo el número de capas ocultas y la cantidad de unidades por capa.

Estas características se establecen a través de un archivo de configuración en formato YAML, el cual se procesa por OpenNMT durante el proceso de entrenamiento. Se puede encontrar ejemplos de las configuraciones utilizadas en el anexo del reporte.

Durante el desarrollo de este proyecto se construyeron modelos con diferentes características. En este capítulo se procederá a explicar los elementos considerados mientras que en el capítulo 5 se explorarán los resultados experimentales a modo de determinar qué modelos obtuvieron los mejores resultados.

4.1. Corpus AnCora

AnCora es un conjunto de corpus creado por el Centro de Lenguaje y Computación (CLIC, *Centre de Llenguatge i Computació*) de la Universitat de Barcelona, disponible tanto en español (castellano) como en catalán. En ambos casos, los corpus se componen de textos periodísticos con cerca de 17 000 oraciones totalizando 500 000 palabras cada uno. El corpus AnCora puede describirse como

un corpus anotado multinivel ya que cada elemento se encuentra manualmente preetiquetado indicando estructura tanto a nivel de cada vocablo (registrando su lexema base, accidentes gramaticales, rol sintáctico y, en algunos casos, roles semánticos) así como la composición de la frase en un árbol estructural. Resulta de especial relevancia en este proyecto la inclusión de marcas que indican la posición esperada de un sujeto implícito así como la categorización de cada palabra según su categoría léxica (*part of speech*, POS). (Delor, Martí, y Recasens, 2008a)

Este corpus se encuentra disponible públicamente para su descarga, presentándose en un formato XML que permite representar la estructura del árbol sintáctico de cada frase y los atributos de cada elemento tanto para grupos sintácticos como nodos terminales (usualmente palabras aunque también se incluyen marcas como las empleadas para indicar un sujeto omitido). (Delor, Martí, y Recasens, 2008b) Para entrenar los modelos secuencia-a-secuencia propuestos en este trabajo se optó por transformar esta representación del corpus en una estructura lineal más simple y más cercana al texto en lenguaje natural sin preprocesar, como se detallará en la sección 4.2.

4.1.1. *POS-tagging* en corpus AnCora

Uno de los metadatos incluidos en el corpus AnCora es la clase gramátical o POS (*part of speech*) correspondiente a cada palabra. Estas comprenden las clases usualmente reconocidas como sustantivos, verbos, adjetivos y preposiciones así como marcas para clasificar elementos que no corresponden propiamente a palabras como los signos de puntuación, fechas y números (incluyendo cantidades monetarias). Cada categoría se codifica a través de una letra como se muestra en el cuadro 4.1. (Delor, Martí, y Recasens, 2008c)

De esta manera, la secuencia de etiquetas correspondientes a las palabras del enunciado *"El niño pequeño comió algodón de azúcar."* será ***"d n a v n s n f"***.

Se debe notar que el corpus en algunos casos trata un grupo de palabras como un solo elemento, especialmente cuando se trata de nombres de entidades como "Real Academia Española", En estos casos se tiene una sola etiqueta, *real_academia_española* se toma como un solo token de clase **n** (sustantivo) en vez de una secuencia de un adjetivo (*real*, **a**), un adjetivo (*academia*, **n**) y un segundo adjetivo (*española*, **a**).

Para este estudio, las etiquetas de clase gramatical se ampliaron con distinciones que, si bien no están contempladas en las etiquetas POS de una letra de AnCora, en general pueden deducirse a partir de otros atributos anotados en el corpus. Esto incluye la discriminación entre verbos finitos y no finitos (ambos etiquetados como **v** en AnCora pero con un tratamiento fundamentalmente distinto a los efectos del proyecto) y las marcas de sujeto omitido (representados en AnCora como un sintagma nominal vacío sin una etiqueta POS correspondiente).

Categoría	Ejemplos	Representación
Adjetivo	*grande, amarilla, repentino*	**a**
Conjunción	*y, pero, como*	**c**
Determinante	*el, unas, aquel*	**d**
Puntuación	*. , (,)*	**f**
Interjección	*ah, ay, oh*	**i**
Sustantivo	*cosa, personas, ejemplo*	**n**
Pronombre	*yo, le, quién*	**p**
Adverbio	*felizmente, recién, bien*	**r**
Preposición	*de, sobre, para*	**s**
Verbo	*cantar, corro, escribiéndole*	**v**
Fecha	*25 de abril, 28/02/2021, 2023*	**w**
Número	*23, 3,14, € 199,90*	**z**

Tabla 4.1: Códigos para las clases gramaticales reconocidas en AnCora. En este proyecto las etiquetas **i** y **w** (originalmente correspondientes a interjecciones y fechas) fueron modificadas a **j** y **k** respectivamente para liberar a **i** y **w** para otros usos (verbos no-finitos y verbos con sujeto omitido).

4.1.2. Caracterización de verbos en el corpus

Las oraciones en el corpus AnCora presentan anotaciones que en la mayoría de los casos permiten caracterizar los verbos de acuerdo a la presencia, omisión o ausencia de sujeto, lo cual lo hace un corpus atractivo para la tarea propuesta en este estudio. Esta información, sin embargo, se encuentra dispersa entre múltiples elementos y atributos, por lo que resultó necesario realizar un preprocesamiento para extraer estas características y así generar el corpus paralelo requerido para este estudio.

En su forma original, el corpus se compone de 1635 archivos XML correspondientes a "documentos" compuestos por múltiples oraciones (típicamente tomadas de un mismo artículo periodístico). Estos archivos XML codifican una estructura arbórea jerárquica donde los nodos representan oraciones, grupos sintácticos (o, en el caso de los signos de puntuación, ortográficos) y lexemas, donde la disposición de los nodos en cada oración corresponde al análisis sintáctico de la misma bajo una gramática de constituyentes. Cada nodo está etiquetado de acuerdo a su categoría (el tipo de sintagma en el caso de grupos sintácticos, la clase gramatical para lexemas) y contiene una serie de atributos que, dependiendo del tipo de nodo, pueden incluir identificadores únicos, información sobre la morfología de un lexema (como persona, número, tiempo, aspecto y modo de un verbo), la forma canónica del lema (el infinitivo de un verbo, la forma masculina singular de un adjetivo, etc), marcas de la función sintáctica y semántica de un grupo nominal entre otros. La figura 4.1 presenta un ejemplo simplificado de esta estructura.

El proyecto requiere la distinción entre verbos no finitos (infinitivos, participios y gerundios; para los que no aplica el reconocimiento de sujeto) y finitos

```
<?xml version='1.0' encoding='utf-8'?>
<article lng='es' id='article000'>
  <sentence id='sentence0001'>
    <S id='S00001'>
      <sn id='sn00001' tem='agt' func='suj' arg='arg0'>
        <spec num='s' gen='m' id='spec00001'>
          <d wd='la'  pos='da0fs0' lem='el' num='s' id='d00001' gen='f'/>
        </spec>
        <grup.nom num='s' gen='f' id='grup.nom00001'>
          <n wd='profesora' pos='ncfs000' lem='profesor' num='s' id='n00001' gen='f'/>
        </grup.nom>
      </sn>
      <grup.verb id='grup.verb00001'>
        <v wd='escribe' lem='escribir' pos='vmip3s0' id='v00001'/>
      </grup.verb>
    </S>
    <conj id='conj00001' conjunctiontype='coordinating'>
      <c id='c00001' postype='coordinating' wd='y' lem='y' pos='cc'/>
    </conj>
    <S id='S00002'>
      <sn elliptic='yes' func='suj' arg='arg0' id='sn00002' tem='agt'/>
      <grup.verb id='grup.verb00002'>
        <v wd='explica' lem='explicar' pos='vmip3s0' id='v00002'/>
      </grup.verb>
    </S>
    <f id='f00001' punct='period' wd='.' lem='.' pos='fp'/>
  </sentence>
</article>
```

Figura 4.1: Ejemplo simplificado del formato de los archivos de AnCora, correspondiente a un documento que solo contiene la oración *"La profesora escribe y explica."*. Obsérvese que el primer elemento ***sn*** (sintagma nominal) corresponde a "La profesora" y contiene el atributo *func='suj'* que lo identifica como sujeto (de "escribe") mientras que el segundo elemento ***sn*** no tiene nodos internos y presenta el atributo *elliptic='yes'*, marcando un sujeto omitido para "explica".

(formas "conjugadas" que pueden presentar un sujeto), así como la caracterización de estos últimos según la presencia de un sujeto explícito, de sujeto omitido o de un uso impersonal. Para esto se analizan atributos contenidos en los nodos correspondientes a los verbos así como los de los grupos nominales dentro de un mismo grupo oracional. La clasificación se basa en las siguientes observaciones:

- Los nodos de verbos contienen un atributo *pos* con una etiqueta que, además de indicar su clase gramatical (*part of speech*, POS), indica las categorías gramaticales de la conjugación verbal como persona, número, tiempo verbal, etc. El contenido de esta etiqueta permite distinguir formas finitas de no finitas.

- Los nodos de sintagmas nominales pueden contener un atributo *func* cuyo valor indica la función sintáctica del grupo. Si el mismo es *'suj'*, se identifica al grupo nominal como sujeto de la oración.

- AnCora incorpora grupos nominales vacíos (sin nodos hijos correspondientes a lexemas) marcados con el atributo *elliptic* para marcar la mayor parte de los sujetos omitido.

- Los grupos verbales (nodos con la etiqueta *grup.verb*) correspondientes a verbos impersonales en oraciones reflejas (con el pronombre clítico "se") suelen presentar el atributo *func='impersonal'*, indicando un uso impersonal.

La consideración de estos atributos permite caracterizar la inmensa mayoría de los verbos en el corpus, aunque existe un 4 % (aproximadamente 1000 verbos finitos) que no presenta marcas correspondientes a ninguna de estas tres categorías por lo que fue necesario etiquetarlos en forma manual. Entre los verbos etiquetados manualmente se identificaron verbos impersonales no contenidos en oraciones reflejas, así como verbos personales con sujeto explícito u omitido pero que no presentaban las marcas correspondientes, presumiblemente debido a errores o criterios dispares en la confección del corpus. El etiquetado manual de estas 1000 entradas permitió generar un corpus más completo para la tarea.

Los criterios de clasificación se explican con mayor detalle en el tercer anexo del informe.

Las anotaciones de AnCora presentan ciertas particularidades en cuanto su criterio para la identificación de sujetos explícitos y omitidos:

- En las oraciones relativas donde el *antecedente* (el argumento común a la cláusula externa) opera como sujeto, se identifica al pronombre relativo *que* o *quien* como sujeto. Por ejemplo, en *"Formaba parte del equipo **que ganó**."*, se identifica al pronombre *"que"* como sujeto explícito del verbo *"ganó"*.

- Cuando existe una conjunción ("y", "o", "sino", etc) entre dos verbos que comparten un sujeto, en AnCora se considera que el sujeto aplica únicamente para el primer verbo, mientras que se considera que el segundo

presenta sujeto omitido. Por ejemplo, en *"La profesora escribe **y** explica."*, donde se podría considerar que "la profesora" opera como sujeto para ambos verbos, se etiqueta a *"la profesora"* como sujeto para únicamente para el verbo *"escribe"* mientras que se anota un sujeto omitido para *"explica"*, efectivamente analizando a la oración como equivalente a *"La profesora escribe y (**ella**) escribe."*.

Para asegurar la consistencia del corpus, estos criterios se replicaron también dentro de las instancias anotadas manualmente para este proyecto.

4.1.3. Partición de corpus

El objetivo de un entrenamiento supervisado es lograr que el modelo a entrenar reconozca patrones relevantes cuyo poder predictivo aplique para instancias no contempladas dentro de los datos de entrenamiento; de otra forma se corre el riesgo de que el modelo pueda explotar relaciones espurias presentes en los ejemplos puntuales con los que se entrenó pero que no generalizan a otros casos, un fenómeno conocido como *overfitting*. Para evitar este problema resulta crucial evaluar la performance del modelo utilizando un conjunto de datos diferentes a los utilizados durante el entrenamiento. Se debe considerar además la existencia de dos instancias de evaluación: una realizada durante el entrenamiento para evaluar el efecto de los hiperparámetros entrenados (lo cual permite además pausar el entrenamiento una vez que el modelo deje de presentar un progreso en la performance, una técnica denominada *early stopping*) y una instancia de evaluación del modelo una vez finalizado el entrenamiento; utilizar conjuntos de datos distintos para esta segunda instancia permite controlar la posibilidad de que el modelo presente *overfitting* respecto a los datos utilizados durante las fases de evaluación del entrenamiento. (Brownlee, 2017)

Esto requiere la partición del corpus en tres segmentos: un subcorpus de entrenamiento (denominado *training set*) a utilizarse durante las fases de entrenamiento del modelo (por ejemplo rondas de *backpropragation*, un subcorpus de validación o desarrollo (*development set* o *validation set*) a emplear en instancias de evaluación durante el entrenamiento y, finalmente, un subcorpus de evaluación (*test set*) para evaluar la performance final de los modelos una vez entrenados. (Brownlee, 2017)

Siguiendo el mismo criterio utilizado en trabajos anteriores del grupo PLN de la Facultad de Ingeniería, se utilizó una partición del corpus AnCora en la que aproximadamente un 80 % de las oraciones se utilizaron como *training set*, un 10 % como *development set* y un 10 % como *testing set*, compuestos de la manera indicada en la siguiente tabla. Únicamente se contabilizan oraciones con al menos un verbo finito (es decir, un verbo conjugado, excluyendo infinitivos, gerundios y participios para los que el problema del sujeto omitido no aplica). La distribución de ejemplos se muestra en el cuadro 4.2.

Partición	Oraciones	Tokens	Verbos finitos	Suj. explícito	Suj. omitido	Impersonales
Entrenamiento	13 489	414 334	34 925	25 545 (73,1 %)	8 835 (25,3 %)	545 (1,6 %)
Desarrollo	1 567	48 768	4 018	2 906 (72,3 %)	1 062 (26,4 %)	50 (1,2 %)
Evaluación	1 623	49 018	4 162	3 043 (73,1 %)	1 062 (25,5 %)	57 (1,4 %)

Tabla 4.2: Partición del corpus AnCora empleada

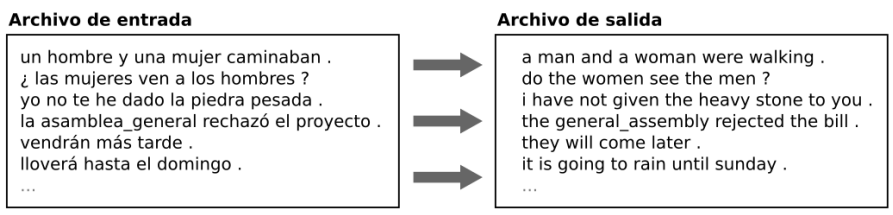

Figura 4.2: Ejemplo de archivos de entrada y salida para un modelo OpenNMT para traducción automática de español a inglés.

4.2. Formato de entrada del modelo

Para el entrenamiento y evaluación de un modelo construido con OpenNMT se deben definir un archivo con las secuencias de tokens de entrada y otro con las secuencias de tokens de la correspondiente salida esperada. Estos archivos deben componerse de una serie de instancias separadas por saltos de línea ($\backslash n$) donde cada instancia está formada por una secuencia de *tokens* separados espacios. La correspondencia entre entradas y salidas esperadas usualmente se maneja a través del número de línea (la línea i del archivo de entradas corresponde a la línea i del archivo de salidas) como se ilustra en la figura 4.2.

En el archivo de configuración de OpenNMT para el entrenamiento del modelo se deberá incluir archivos de entrada y salida para utilizar como corpus de entrenamiento (donde se utilizaron las instancias de la partición de entrenamiento de AnCora) así como archivos de entrada y salida para utilizar en la validación del entrenamiento (instancia de la partición de desarrollo de AnCora). El formato utilizado en los archivos de entrada y salida durante el entrenamiento determina el formato de entrada esperado para el modelo una vez entrenado así como la salida generada.

En el caso de este proyecto, el formato de entrada contiene leves desviaciones respecto a un texto ordinario en español debido a características propias de OpenNMT (como el requisito de separar los tokens mediante espacios) así como por particularidades en la segmentación en palabras empleada por AnCora. Debe notarse que el formato empleado no utiliza información previa sobre la estructura del enunciado u otras características reconocidas en AnCora, únicamente el listado de palabras y signos ortográficos que componen la oración de acuerdo al criterio empleado por el corpus. Las principales desviaciones respecto a un texto en español sin procesar son las siguientes:

- No distinción entre mayúsculas y minúsculas; toda la entrada se convierte

a minúsculas. Al evitar que una misma palabra pueda representarse mediante tokens distintos según el uso de mayúsculas se busca que el modelo obtenga una mayor capacidad de generalización.

- Las marcas de puntuación (incluyendo puntos, comas, signos de interrogación, paréntesis, etc.) se consideran como tokens individuales por lo que se separan por espacios de las palabras contiguas: "¿Cómo estás?" se transforma en ‘’¿ cómo estás ?“.
- Siguiendo el criterio de AnCora, ciertos nombres compuestos por múltiples palabras se tratan como un solo token, utilizando el caracter de guion bajo para unir las palabras individuales. Esto aplica tanto para nombres como "juan_pérez" por "Juan Pérez" como para determinadas frases formadas a partir de sustantivos comunes como "real_academia_española" por "Real Academia Española". Aplica un criterio similar para grupos correspondientes a números o fechas: "14_de_febrero" en vez de "14 de febrero".

4.3. Formatos de salida del modelo

Para la consecución del objetivo previsto, la salida de los modelos de traducción deberá permitir la clasificación de cada verbo finito en las categorías consideradas, ya sea como una clasificación binaria entre verbo con o sin sujeto explícito o como una clasificación ternaria entre verbo con sujeto explícito, verbo con sujeto omitido o verbo impersonal. Durante el desarrollo de este proyecto se abordó en una primera instancia el problema de clasificación binaria extendiendo posteriormente los resultados al caso ternario.

Para la elección de un formato de salida a utilizar se consideró el precedente de *Grammar as a Foreign Language* (Vinyals y cols., 2014), donde la salida generada por el modelo incluye tanto tokens correspondientes a las palabras de la entrada como tokens especiales que indicaban la posición de los elementos gramaticales que dicho trabajo buscaba detectar, como el inicio y el fin de determinados grupos sintácticos. El formato a utilizar en la salida repercute directamente en el vocabulario de salida del modelo, es decir, en el conjunto de opciones posibles que puede tomar cada token generado por la salida. En un modelo como el planteado en dicho artículo, si se parte de un vocabulario de entrada (opciones posibles para los tokens de entrada, usualmente entre 10^4 y 10^5 opciones) de tamaño N y se cuenta con m etiquetas adicionales para indicar la presencia de características gramaticales, la dimensión o número de elementos del vocabulario de salida será de $N + m$. Esta dimensión determina el número de unidades en la capa de salida (*output layer*) del modelo, por lo que un mayor vocabulario conlleva un incremento en el número de parámetros a entrenar así como en la cantidad de memoria necesaria para el entrenamiento y ejecución del modelo.

En una fase preliminar se identificó que los recursos de hardware utilizados (especificados en Anexo 1: Características del hardware empleado), en particular la cantidad de memoria dedicada en la GPU (4 GB), resultaba insuficiente para

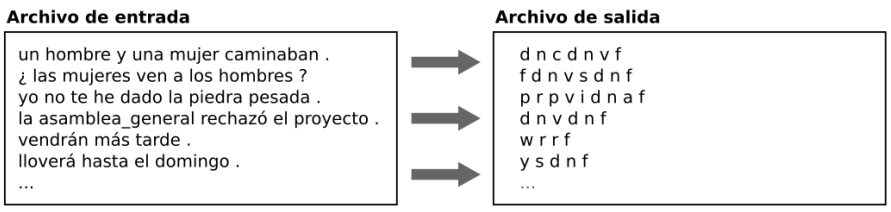

Figura 4.3: Ejemplo de archivos de entrada y salida para uno de los modelo OpenNMT con *POS-tagging* modificado para la detección de sujetos omitidos, en este caso utilizando una etiqueta v para verbos con sujeto explícito, w para verbos con sujeto omitido o y para verbos impersonales.

el entrenamiento de un modelo donde la dimensión del vocabulario de salida fuera de un orden similar a la dimensión del vocabulario de entrada (compuesto por 38724 palabras). Se observó asimismo que recuperar las palabras de la entrada no resultaba particularmente relevante para el problema tratado donde basta con determinar la posición aproximada del sujeto. Estos factores llevaron a adoptar la decisión de utilizar una salida basada en las clases gramaticales (*part of speech*, POS) de los elementos en la entrada, lo cual permite reducir el vocabulario de salida a solo una docena de elementos incluyendo las clases POS etiquetadas en el corpus AnCora y marcas auxiliares para el reconocimiento de objetos omitidos, lo cual posibilitó el entrenamiento de los modelos con recursos de hardware más limitados.

Esto lleva a que, en efecto, los modelos propuestos sean etiquetadores de clase gramatical (POS-taggers), aunque con modificaciones para tratar la clasificación de frases verbales según la presencia o ausencia de sujeto omitido. Cada clase gramatical se representa a través de su código de una letra en AnCora, como se presentó en la sección 4.1.1. A modo de ejemplo, la oración "El hombre compra pan." se transformará en la secuencia "d n v n f", con las etiquetas de AnCora correspondientes a un determinador **d** (el artículo "el"), un sustantivo **n** ("hombre"), un verbo **v** ("compra"), otro sustantivo **n** ("pan") y una marca de puntuación (el punto final). La figura 4.3 presenta ejemplos de pares entra-salida utilizando un criterio como el planteado.

Dado que el problema de *POS-tagging* resulta meramente accesorio en estos modelos, sus resultados no se toman en cuenta al evaluar los modelos; para una entrada dada únicamente resultará relevante si el modelo fue capaz de determinar la presencia de sujeto omitido o de oraciones impersonales para cada grupo verbal. Sin embargo, la performance del modelo como etiquetador resulta un dato relevante durante la fase de entrenamiento ya que supone parte del objetivo para el cual OpenNMT se optimiza el modelo. Esto lleva a que la tasa de acierto en *POS-tagging* constituya un indicador del progreso o estancamiento del modelo durante su entranimiento. Se debe considerar, sin embargo, que un aumento en la tasa de acierto de este problema auxiliar no necesariamente correspondrá a un aumento en la performance del problema de clasificación

central o viceversa.

La salida correspondiente a cada frase del corpus se construirá a partir de la secuencia de etiquetas de POS con modificaciones orientadas a la detección de los verbos a clasificar (generando una distinción entre los verbos finitos para los que aplicará la clasificación y los verbos no-finitos para los que no lo hará) así como nuevas marcas para manejar el problema de clasificación en sí, para lo cual se emplea información comprendida dentro del corpus AnCora incluyendo anotaciones sobre sujetos omitidos y atributos que señalan un verbo como impersonal.

Se consideraron dos enfoques respecto al tratamiento de estas marcas especiales: utilizar un token para indicar la posición de un sujeto omitido o utilizar diferentes tokens para los verbos según su sujeto.

Modificaciones comunes a ambos enfoques

En el corpus AnCora se utiliza la etiqueta **v** para indicar la clase gramatical de todos los verbos independientemente de su uso, lo cual comprende tanto verbos finitos (capaces de gobernar una frase verbal u oración) como verbos no finitos (infinitivos, gerundios y participios, incapaces de formar oraciones por sí mismos). Como se explica en el capítulo de Conceptos, el problema de reconocimiento de sujeto omitido entendido como clasificación de frases verbales únicamente aplica para verbos finitos, debiéndose emitir tantas clasificaciones como verbos finitos contenga la entrada a evaluar.

Esto lleva a que resulte necesario distinguir en la salida entre verbos finitos y no-finitos, pese a que técnicamente se engloben dentro de una misma clase léxica. Para esto se procedió a emplear la etiqueta **i** para los verbos no finitos. En caso de construcciones con uno o más verbos auxiliares, únicamente el primer verbo auxiliar aparecerá en una forma finita por lo que se lo marcará de la forma usual (**v**) mientras que los demás verbos que intervengan aparecerán en formas no-finitas por lo que se los podrá etiquetar con **i**:

- "La niña está cantando." → **d n v i f**
- "Él lo iba a hacer." → **p p v s i f**
- "Tú habías estado intentado solucionar el problema ." → **p v i i i d n f**

En estos casos el problema de clasificación únicamente a ese primer verbo auxiliar ("está", "iba", "habías'), lo cual resulta apropiado desde un punto de vista sintáctico ya que son estos verbos los que presentan inflexiones que pueden corresponder a un sujeto, pese a que el rol que cumple dicho sujeto a nivel semántico esté dado por el verbo final (el agente que "canta", "hace" y "soluciona"). Dado que el sujeto es un concepto puramente sintáctico, las consideraciones semánticas no tienen incidencia en la clasificación.

Se optó por modificar la representación utilizada en AnCora para las interjecciones y las fechas (etiquetadas como **i** y **w**) por **j** y **k** para liberar las etiquetas *i* y *w* para otros usos (verbos no finitos y verbos con sujeto omitido en el enfoque de clasificación de verbos).

Enfoque de marca de sujeto omitido

Las instancias que muestran un sujeto omitido (incluyendo los casos de oraciones impersonales) se notan en el corpus AnCora mediante un elemento dentro del árbol de parsing que representa a este sujeto omitido, donde la ubicación de dicho elemento dentro del árbol corresponde a la posición en la que se encontraría el sujeto en caso de haberse explicitado según el criterio de los anotadores humanos que desarrollaron el corpus. El enfoque de "marca de sujeto omitido" plantea incorporar a la salida un token **0** (cero) que indique la posición de estas marcas. De este modo, mientras que la oración con sujeto explícito "Juan duerme." tendrá la salida "n v f" (sustantivo, verbo, puntuación), la oración con sujeto implícito "Duerme." tendrá por salida '0 v f' (sujeto omitido, verbo, puntuación).

- "Juan duerme." → **n v f**
- "Duerme." → **0 v f**

Un análisis del corpus revela que las marcas de sujeto omitido predominantemente se ubican precediendo inmediatamente al verbo (por lo que un verbo con sujeto omitido usualmente se marcará en la salida la secuencia **0 v**) aunque existen excepciones que pueden catalogarse como inconsistencias al encontrarse criterios dispares ante estructuras sintáctica y semánticamente idénticas. Entre los casos excepcionales encontrados se encuentran:

- Posicionamiento de marcas cuando el verbo es precedido por pronombres preclíticos: "me lo dijo" puede anotarse como **p p 0 v** (siguiedo el criterio general de emplear la marca **0** antes del verbo) o como **0 p p v** (precediendo a los pronombres, en la ubicación natural para un sujeto).
- Posicionamiento cuando el verbo es precedido por adverbios como "no", "aún" o "todavía"; "aún no opinó" puede anotarse como **0 r r v**, **r 0 r v** o como **r r 0 v**.
- Combinaciones de los casos anteriores ("no lo sabe" como **0 r p v**, **r 0 p v** o **r p 0 v**).
- Casos en los que la marca de sujeto omitido se coloca antes de acotaciones entre paréntesis o comas que preceden al verbo: "en esta versión, además, cuenta que..." analizado como **s d n 0 f r f v c**, ubicando al hipotético sujeto entre "versión" y la coma subsiguiente en vez del análisis más esperable **s d n f r f 0 v c** con el sujeto precediendo al verbo.
- En un número limitado de oraciones imperativas o interrogativas la marca de sujeto se ubica a continuación del verbo ("¡Corre!" como **f v 0 f**, "¿Qué buscó en él?" como **f p v 0 s p f**) o, en algunos casos, al final de la oración ("¿Qué buscó en él?" analizado como **f p v s p 0 f**) aunque ambos tipos de oraciones usualmente se anotan ubicando la marca antes del verbo ("¡Corre!" como **f 0 v f**, "¿Qué buscó en él?" como **f p 0 v s p f**).

Para el problema de clasificación no resulta relevante la posición en la que el modelo genera el token **0** (de haberlo), solamente que el mismo pueda identificarse como correspondiente al verbo respectivo. A modo de ejemplo, si la entrada "No se lo dará." genera la salida **r p p 0 v f** (detectando un sujeto omitido previo al verbo "dará"), se considerará la detección como correcta pese a que la salida esperada pudiera estar etiquetada con la marca al principio como **0 r p p v f**.

En el caso de que una instancia contenga más de un verbo finito, resultará necesario determinar a qué verbo corresponde una marca **0**. A modo de ejemplo, considérese el enunciado "Él cocina y está condimentando los platos." donde se tiene dos grupos verbales unidos por una conjunción: "él cocina" (donde el verbo finito "cocina" presenta un sujeto explícito, **p v**) y "está condimentando los platos" (donde el verbo finito "está (condimentando)" presenta sujeto omitido aunque se lo pueda inferir de la primera mitad de la frase, **0 v i d n**). Una salida correspondiente a una clasificación correcta podría ser **p v c 0 v i d n f** aunque para su evaluación se hace necesario aplicar un criterio que determine si la marca **0** corresponde a un sujeto omitido para el primer verbo ("cocina") o para el segundo ("está"). Esta situación se resuelve mediante las siguientes heurísticas, basadas en la distribución encontrada en el corpus:

- Una marca **0** seguida por un verbo **v** se asocia a dicho verbo.
- Una marca **0** seguida por pronombres (**p**) y/o adverbios (**r**) y posteriormente un verbo **v** corresponde a dicho verbo.
- En otros casos, la marca **0** se asociará al verbo **v** más cercano, priorizando al verbo posterior en caso de empate.

Si bien estas heurísticas podrían dar lugar a clasificaciones incorrectas, se comprobó que predicen correctamente la correspondencia entre marcas de sujeto omitido y verbos para todas las entradas incluidas en el corpus.

El eventual caso de que la salida del modelo genere más de una marca **0** para un verbo dado (por ejemplo **0 p 0 v f** para "Lo hizo.", ubicando el sujeto implícito tanto al inicio como previo al verbo; una inconsistencia no encontrada en el corpus) se considerará equivalente a la presencia de una sola marca.

Enfoque de clasificación de verbos

El uso de un token **0** en la salida correspondiente a la posición hipotética de un sujeto omitido presenta las desventajas de que la marca puede generarse en diferentes posiciones para un verbo dado y que puede haber ambigüedades en la determinación de a qué verbo corresponde una marca dada en caso de que la instancia contenga más de un verbo finito. Ambas dificultades pueden resolverse prescindiendo del uso de un token independiente para indicar el sujeto omitido y, en cambio, emplear etiquetas distintas para el verbo según este presente o no un sujeto explícito.

Bajo este enfoque, cada verbo finito en la salida utilizará una etiqueta (**v**) en caso de presentar un sujeto explícito y una etiqueta distinta (**w**) en caso de que

presente sujeto omitido (independientemente de la posición en la que se note esta característica en el árbol de parsing de AnCora).

- "Juan duerme." → **n v f**
- "Duerme." → **w f**

Este enfoque resulta más natural si se entiende el problema de detección de sujeto omitido como un problema de clasificación donde cada verbo finito se clasificará como **v** (sujeto explícito) o **w** (sujeto omitido).

Oraciones impersonales

Los primeros modelos construidos en el marco de este estudio consideraron la presencia o ausencia de un sujeto explícito como una distinción binaria, ignorando una posible distinción entre oraciones con un sujeto omitido que podría haberse incluido en forma explícita (como en "Lee un libro", donde se podría explicitar un sujeto como "Ella lee un libro" o "El alumno lee un libro") y las oraciones impersonales cuya sintaxis no admite la incorporación de un sujeto explícito (como en "Llueve" o en ciertas oraciones reflexivas como "Se cree que es así"). El abordaje del problema como una clasificación binaria fue motivado por el tratamiento que se le da a los sujetos implícitos en AnCora, el corpus utilizado, donde se incluyen marcas de sujeto implícito tanto para las oraciones impersonales como para las personales que presentan propiamente sujeto omitido. El corpus, sin embargo, registra la distinción entre oraciones personales e impersonales de un modo independiente (mediante un atributo en los nodos correspondientes a la raíz de un grupo verbal en el árbol de parsing), lo cual hace posible reconocer las tres categorías para las instancias obtenidas del corpus y construir modelos de clasificación ternaria comparables a los construidos en los trabajos mencionados como antecedentes.

Como se detalla en *Resultados experimentales*, los experimentos orientados a la clasificación binaria demostraron que los modelos construidos con el enfoque de clasificación de verbos (utilizar tokens **v** y **w** para verbos finitos con y sin sujeto explícito, respectivamente) presentan un rendimiento marcadamente superior a los construidos con el enfoque de marca de sujeto omitido (usar **v** para todos los verbos finitos y **0** para la posición hipotética de un sujeto implícito), por lo que se planteó abordar la clasificación en tres categorías a través de un enfoque que utilice tokens distintos para las mismas: **v** para verbos finitos con sujeto explícito, **w** para verbos finitos no-impersonales con sujeto omitido y una nueva etiqueta **y** para verbos finitos utilizados en oraciones impersonales. A modo de ejemplo, considérense la clasificación del verbo "duerme" en los siguientes enunciados:

- "Juan duerme bien en el hotel." → **n v r s d n f** (sujeto explícito, **v**)
- "Duerme en el hotel." → **w s d n f** (sujeto omitido, **w**)
- "Se duerme bien en el hotel." → **p y r s d n f** (oración impersonal, **y**)

Puede observarse que el último ejemplo, “Se duerme bien en el hotel.”, admite también una interpretación personal: que una persona consigue dormirse (“se duerme”) en una forma profunda (“bien dormido”) aunque ante la ausencia de contexto resulta más razonable interpretar la oración como una oración reflexiva impersonal en la que “se duerme bien” no aplica a un sujeto específico sino que enuncia una verdad genérica sobre el modo en el que cualquier eventual huésped dormiría en el hotel. Esto resulta ilustrativo de un problema que resulta común a muchas oraciones impersonales: la posibilidad de que ante diferentes contextos una misma secuencia admita una interpretación como construcción personal y otra como impersonal. Dado que los modelos secuencia-a-secuencia planteados se entrenan a modo de procesar un enunciado a la vez, sin contar con un contexto a nivel de párrafo o de discurso, esto podría llevar a clasificaciones erróneas.

Verbos no reconocidos y verbos espurios

En los enfoques planteados, la clasificación de cada verbo finito en un enunciado a analizar se lleva a cabo a partir de la secuencia de tokens de POS generada por el modelo secuencia a secuencia de OpenNMT; la obtención de una clasificación para cada cláusula se da en dos etapas: la traducción del formato de entrada al formato de salida a través del modelo entrenado y la interpretación de la secuencia de salida obtenida a través de un algoritmo fijo, definido por el enfoque adoptado. Debe tenerse en cuenta que la salida del modelo de aprendizaje automático es falible, lo cual puede llevar no solo a que existan salidas cuya interpretación arroje un resultado erróneo sino a que puedan existir salidas que no admitan una interpretación por no poseer la estructura adecuada. Esto ocurre cuando el número de verbos finitos en la salida del modelo (para los cuales se puede computar una clasificación) difiere del número de de verbos finitos en la entrada y la salida esperada, ya sea porque el modelo reconoció menos verbos finitos que los realmente presentes (por lo que se tienen verbos no reconocidos para los que no se ofrece una clasificación) o porque se reconocieron más verbos finitos que los existentes en la frase (verbos espurios cuya clasificación no corresponde a un valor real).

El caso de obtener una salida con un M verbos finitos siendo que la salida esperada muestra N verbos finitos con $N > M$ implica que hubo al menos $n-m$ verbos a clasificar en la entrada que no fueron reconocidos como tales por el modelo y para los que no se obtiene una clasificación. Esto puede deberse a que los atributos del modelo no se encuentran suficientemente entrenados por lo que se obtiene una salida azarosa o ser el resultado de una ambigüedad léxica como en el siguiente ejemplo:

- “Mide el largo de la tabla y la pesa.” → **w d n s d n c p w f** - “mide” y “pesa” son verbos con sujeto omitido, correctamente etiquetados como **w**
- “Mide el largo de la tabla y la pesa.” → **w d n s d n c d n f** - se toma “pesa” como sustantivo (**n**, como un segundo elemento cuyo largo es medido por el sujeto de “mide”), el resultado permite reconocer correctamente

la categoría de “mide” pero no clasifica a “pesa” en ninguna de las tres categorías definidas (**v** para sujetos explícitos, **w** para sujetos omitidos o **y** para impersonales)

Esto impacta directamente en los resultados obtenidos por los modelos planteados en este trabajo ya que requiere considerar un resultado adicional posible para la clasificación de las instancias: oración no clasificada por no detección de verbo. En el caso de los modelos de clasificación binaria (sujeto explícito vs ausencia de sujeto explícito) en realidad resulta posible obtener tres resultados (“con sujeto explícito”, “sin sujeto explícito” o “no detectado”) mientras que en la clasificación ternaria se obtienen cuatro tipos de resultado (“con sujeto explícito”, “con sujeto omitido”, “oración impersonal” o “no detectado”). Dado que la no-detección jamás representa un valor esperado, los resultados del modelo pueden expresarse con una matriz de confusión de dimensiones $n \times (n+1)$ en vez de $n \times n$ al haber n categorías de entrada y $n+1$ categorías de salida. Esta particularidad afecta el cálculo de las métricas del modelo, como se examinará en la sección 5.1.

También puede presentarse el caso opuesto, en el que la salida del modelo contenga un número M de tokens de verbo finito a clasificar superior a los N verbos esperados para la instancia, indicando que la salida generó al menos $M - N$ verbos espurios cuya clasificación no tiene valor práctico. Esto usualmente ocurre en modelos con un entrenamiento insuficiente, los resultados experimentales muestran que el número de verbos espurios se reduce a lo largo del entrenamiento. Las clasificaciones correspondientes a verbos espurios pueden ignorarse, aunque para esto resulta necesario identificar entre los M verbos finitos de la salida cuáles corresponden a los N verbos esperados y cuáles a los $M-N$ verbos espurios. A modo ilustrativo, considérese el siguiente ejemplo (ficticio):

- “Se observa que la mayoría cumplió con los objetivos.” → **p y c d n v s d n w f**

En el ejemplo se tiene $N = 2$ verbos finitos: “observa” (utilizado en una oración impersonal, por lo que se espera la etiqueta **y**) y “cumplió” (el cual presenta el sujeto explícito “la mayoría”, correspondiendo la etiqueta **v**) mientras que en la salida se encuentran $M = 3$ tokens que corresponden a verbos finitos: **y** en segunda posición (correspondiente a “observa”), **v** en sexta posición (correspondiente a “cumplió”) y **w** en décima posición (un verbo espurio en la salida). El sistema de clasificación requiere un criterio que determine qué resultados descartar.

El criterio adoptado en este trabajo asume que la posición de los tokens dentro de la secuencia de salida obtenida mantendrá un cierto grado de correspondencia con los tokens en la secuencia de entrada por lo que cada verbo podrá emparejarse con el verbo cuya posición relativa en la secuencia sea más similar. Para cada elemento a emparejar se calcula su posición relativa con un número real entre 0 y 1 donde 0 corresponde al primer token de la secuencia y 1 al último según la siguiente fórmula

Verbos finitos en la entrada/salida esperada

Verbo	Posición	Posición relativa
observa	2° token de 10	0,11
cumplió	6° token de 10	0,55

Verbos finitos en la salida obtenida

Verbo	Posición	Posición relativa
y	2° token de 11	0,1
v	6° token de 11	0,5
w	10° token de 11	0,9

Distancia (absoluta) entre posiciones relativas

	y	***v***	***w***
observa	0,01	0,39	0,79
cumplió	0,45	0,05	0,35

Figura 4.4: Emparejamiento entre los verbos finitos de la oración "Se observa que la mayoría cumplió con los objetivos." y los tokens de verbo finito en la secuencia **p y c d n v s d n w f**

$$posicion\_relativa(x) = \frac{posicion(x) - 1}{\#tokens - 1}$$

Tras obtener las posiciones relativas de cada elemento se puede proceder a construir una matriz de distancias con la diferencia absoluta entre las posiciones relativas de cada emparejamiento posible entre los elementos de cada secuencia. A partir de dicha matriz se puede aplicar un método *greedy* para asignar los emparejamientos, priorizando aquellos que presentan una menor distancia. En la figura 4.4 se aplica este criterio al ejemplo introducido anteriormente. La menor distancia se identifica entre "observa" y el token **y** por lo que se asume una correspondencia entre ellos, descartando los otros potenciales emparejamientos para ambos elementos. Entre los elementos restantes se reconoce que "cumplió" presenta una distancia menor a la etiqueta **v** que a la etiqueta **w**, por lo que se empareja con la primera. La etiqueta **w**, al no poderse emparejar, se identifica como un verbo espurio a descartar.

Este mismo algoritmo también puede utilizarse para determinar cuáles fueron los verbos reconocidos en caso de que el número de verbos identificados en la salida obtenida sea menor al esperado.

Cabe observar que la salida de un modelo para una instancia dada podría presentar simultáneamente verbos no reconocidos y verbos espurios. Considérese el siguiente ejemplo (ficticio):

- "Nosotros bebimos vino que nos había regalado un amigo." → **p v w c p i d n f**

El ejemplo presenta dos verbos finitos a clasificar: "bebimos" (con sujeto explícito, "nosotros") y "había (regalado)" (con un sujeto explícito pospuesto, "un amigo"). El resultado esperado es **p v n c p v i d n f**, con etiquetas **v** para ambos casos. El resultado obtenido, sin embargo, muestra una etiqueta **v** que parece corresponder a "bebimos" (correcta) y una etiqueta **w**, correspondiente a un verbo con sujeto omitido, que posiblemente se originó por interpretar

erróneamente al sustantivo “vino” como el verbo homónimo. Por otro lado, la salida no incluye un token que corresponda al verbo “había”, por lo que se tiene también un verbo no reconocido. Sin embargo, ambos errores se compensan llevando a que el número de verbos finitos en la salida obtenida y en la salida esperada coincida, $M = N = 2$. En este caso, los dos errores no se detectarán y el algoritmo de evaluación considerará que las salidas obtenidas **v** y **w** corresponden a los verbos “bebimos” (clasificado correctamente como **v**) y “había” (que pasará a clasificarse incorrectamente como **w**). Si bien en este ejemplo el error resulta aparente, el hecho de que la salida no codifica información identificativa del verbo original lleva a que en el caso general no sea posible distinguir este tipo de errores de una salida en la que cada verbo obtuvo efectivamente una clasificación válida.

4.4. Word-embeddings

En los modelos construidos con OpenNMT para este trabajo, los tokens recibidos como entrada se representan a través de vectores de 300 dimensiones denominados embeddings de palabras o *word-embeddings*. Como se explicó en la sección 2.2.6, el vector correspondiente a cada palabra del vocabulario puede generarse en forma aleatoria pero también es posible utilizar una colección preexistente de vectores (un embedding preentrenado) donde la representación de cada palabra encapsula información sobre los contextos en los que suele aparecer, a partir de lo cual se pueden recuperar relaciones sintácticas y semánticas, lo cual permite obtener mayores rendimientos en múltiples aplicaciones de PLN.

En el presente trabajo se exploró el efecto de utilizar tres colecciones de embeddings preentrenados en base a textos en idioma español, en adición a los embeddings aleatorios generados por OpenNMT.

- Una colección generada mediante la técnica ***GloVe*** a partir del *Spanish Billion Word Corpus* (SBWC). (Rojas y cols., 2018)
- Una colección generada mediante la técnica ***Word2Vec*** a partir del mismo corpus. (Rojas y cols., 2018)
- La colección **emb39** generada mediante la técnica ***Word2Vec*** a partir de un corpus generado por el Grupo PLN de la Facultad de Ingeniería de la Universidad de la República. (Azzinari y Martínez, 2016)

4.5. Hiperparámetros explorados

Durante el desarrollo del estudio se exploró el rendimiento de modelos con diferentes dimensiones en cuanto al número de capas ocultas en el modelo de OpenNMT así como el número de unidades en cada capa oculta.

Se exploraron modelos con capas ocultas con 100, 500, 1000 y 1500 unidades encontrándose que el hardware utilizado no fue capaz de manejar modelos con

una dimensionalidad mayor. Asimismo, se exploró el uso de 1, 2, 3 o 4 capas ocultas, al detectarse que los resultados con cuatro capas resultaron de inferior calidad a resultados análogos con solo tres capas se consideró innecesario explorar modelos con una mayor profundidad.

Se observó asimismo que la cantidad de pasos de entrenamiento (el número de iteraciones de backpropagation sobre el conjunto de ejemplos en el corpus de entrenamiento) necesaria para que el modelo alcanzara su mayor rendimiento mostró una dependencia respecto a las dimensiones del modelo y el uso de colecciones preentrenadas de embeddings. Inicialmente se estudió la evolución de los modelos a través de 20000 pasos de entrenamiento, evaluando resultados intermedios cada 2000 pasos, encontrándose que esto resultaba suficiente para modelos con hasta 2 capas ocultas de 500 unidades, los cuales presentaron un pico en su rendimiento antes de los 20000 pasos, tras lo cual los resultados se deterioraron, posiblemente a causa de overfitting. Para modelos mayores, el rendimiento de los modelos continuó mejorando más allá de los 20000 primeros pasos, por lo que se realizaron pruebas de hasta 100000 pasos de entrenamiento, evaluando el rendimiento del modelo cada 10000 pasos.

Capítulo 5

Resultados experimentales

Los diferentes enfoques y configuraciones de hiperparámetros presentados en el capítulo 4 fueron evaluados experimentalmente mediante el entrenamiento de modelos y su evaluación sobre el corpus de desarrollo.

Dado el elevado número de combinaciones paramétricas posibles, se optó por explorar el rendimiento al alterar diferentes elementos en fases. En una primera instancia, se evaluó el rendimiento de los dos enfoques considerados para el formato de salida, detectándose una marcada ventaja en el uso del "enfoque de clasificación de verbos" (utilizar tokens **v** o **w** en la salida del modelo de OpenNMT para verbos con o sin sujeto explícito, respectivamente) frente al "enfoque de marca de sujeto omitido" (utilizar un token **0** independiente a la representación del verbo para indicar un sujeto implícito). Tras la obtención de este resultado, se exploró en una segunda instancia el rendimiento de este enfoque con diferentes colecciones de *word-embeddings* y, tras observar un rendimiento superior con el uso de embeddings *GloVe*, una tercera instancia explorando una mayor cantidad de combinaciones de números de capas ocultas y cantidad de unidades por capa en las redes neuronales construidas. En una cuarta y última instancia, se exploró el potencial de los modelos planteados para el problema de clasificación ternaria de oraciones, incorporando la distinción entre oraciones personales con sujeto omitido y oraciones impersonales a modo de obtener resultados comparables con los de los estudios precedentes.

En este capítulo se presentan los principales resultados observados. Se incluye un detalle de los resultados obtenidos por cada modelo en los anexos del informe.

5.1. Efecto de verbos no reconocidos en métricas

Como se explicó en la sección 4.3, el uso de un modelo de traducción falible como paso intermedio para la clasificación de los verbos lleva a que en algunos casos el modelo no reconozca un verbo finito como una instancia a clasificar y, por ende, sea incapaz de generar una salida válida. Esto implica que al considerar

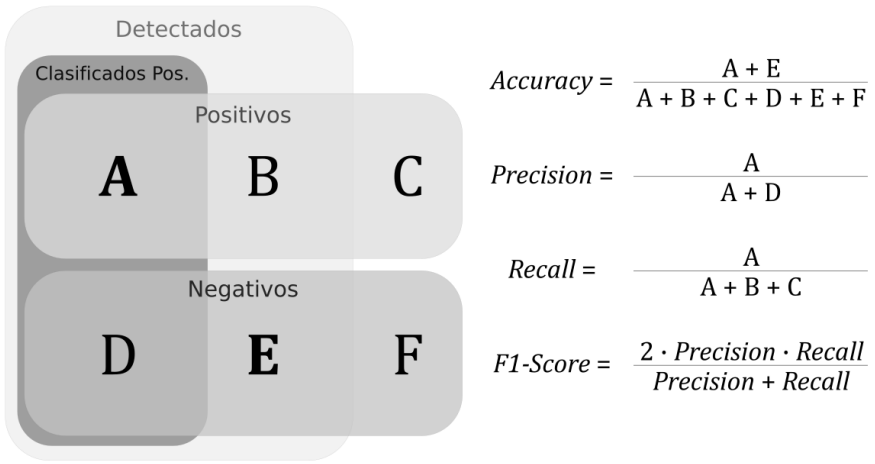

Figura 5.1: Diagrama de resultados posibles de los modelos planteados para la clasificación respecto a una categoría. Las instancias pueden tener resultado esperado positivo (A, B y C) o negativo (D, E y F) pero el modelo puede devolver una clasificación positiva (A y D), devolver una clasificación negativa (B y C) o no detectar el verbo y, por ende, no devolver una clasificación válida (C o F), lo cual se tratará igualmente como un negativo a efectos del cálculo de métrica. Idealmente, ante un clasificador perfecto, todas las instancias deberían volcarse a los conjuntos A (verdaderos positivos) y E (verdaderos negativos, correctamente detectados).

los resultados de clasificación para una clase dada (por ejemplo las instancias con sujeto omitido) no solo se tenga resultados positivos y negativos de la forma usual sino que también se tenga instancias no categorizadas. Dado que estos resultados anómalos no corresponden a ninguna categoría, se los puede tratar como un caso particular de negativo. De esta forma, las instancias para los que el valor esperado es positivo pero que no fueron detectadas por el modelo se contabilizarán como falsos negativos, teniendo el mismo efecto sobre las medidas de *accuracy* y *recall* que las instancias que fueron reconocidas como verbos finitos por el modelo pero que tuvieron una clasificación errónea. Los verbos no reconocidos no afectan el valor de la métrica *precision* ya que la misma concierne únicamente las instancias para las que la salida obtenida fue positiva. Estas situaciones se ilustran en el diagrama 5.1.

Dado que los verbos espurios (detecciones de verbos finitos en la salida del modelo que no corresponden a un verbo en la salida esperada) se descartan en la evaluación, no presentan incidencia sobre las métricas obtenidas.

5.2. Clasificación binaria - presencia o ausencia de sujeto explícito

Los experimentos descritos a continuación buscaron distinguir entre verbos que presentan sujeto explícito y verbos que no lo hacen, ya sea debido a que presentan un sujeto omitido propiamente dicho o a que se trata de un uso impersonal. En la sección subsiguiente se abordan experimentos de clasificación ternaria, incorporando la distinción entre oraciones impersonales que no pueden presentar sujeto y oraciones personales con sujeto omitido.

5.2.1. Evaluación de enfoques a aplicar

Se llevó a cabo una primera instancia de experimentación con el objetivo de probar la factibilidad del concepto de reconocer la presencia o ausencia de sujeto mediante un modelo de traducción en OpenNMT y para determinar cuál de los dos enfoques propuestos para la salida del modelo resultaba más prometedor.

En esta etapa se evaluó además la factibilidad de utilizar modelos con una o dos capas ocultas (en ambos casos de 500 unidades por capa) así como el uso de embeddings preentrenados *GloVe* creada a partir del corpus SBWC (*Spanish Billion Word Corpus*) frente al uso de embeddings aleatorios autogenerados por OpenNMT.

Enfoque	Embeddings	Capas ocultas	TER	Verbos no detectados	Accuracy	Precision	Recall	Medida F1
Marca **0** para sujeto implícito	Aleatorio	1×500	11,57 %	8,24 %	77,38 %	74,04 %	51,53 %	60,77 %
Marca **0** para sujeto implícito	Glove SBWC	1×500	23,97 %	10,80 %	73,65 %	68,05 %	47,04 %	55,62 %
Marca **0** para sujeto implícito	Aleatorio	2×500	7,90 %	5,13 %	81,16 %	72,13 %	64,82 %	68,28 %
Marca **0** para sujeto implícito	Glove SBWC	2×500	8,35 %	5,50 %	82,33 %	**74,87 %**	68,07 %	71,31 %
Marcas **v** y **w**	Aleatorio	1×500	28,70 %	16,37 %	67,73 %	70,06 %	31,55 %	43,51 %
Marcas **v** y **w**	Glove SBWC	1×500	61,71 %	43,44 %	45,01 %	48,97 %	9,08 %	15,32 %
Marcas **v** y **w**	Aleatorio	2×500	4,70 %	**4,68 %**	81,04 %	69,06 %	73,61 %	71,26 %
Marcas **v** y **w**	Glove SBWC	2×500	**4,68 %**	5,92 %	**83,03 %**	73,06 %	**74,67 %**	**73,85 %**

Tabla 5.1: Experimento 1: Mejores métricas logradas por cada modelo durante su entrenamiento, medidas respecto al corpus de desarrollo. Se indica en negrita los mejores valores obtenidos para cada métrica.

Como muestran las métricas en la tabla 5.1, estos experimentos demostraron que la técnica propuesta resultaba viable para el reconocimiento de sujetos omitidos utilizando modelos con al menos dos capas ocultas; los modelos con una sola capa oculta presentaron un rendimiento pobre al ser incapaces de reconocer adecuadamente la estructura de las secuencias a generar (su salida tiende a repetir subsecuencias de tokens). Los resultados para los modelos con dos capas ocultas, ilustrados en el diagrama 5.2 muestran un mejor rendimiento (evaluados según la medida F1) cuando incorporan el uso de la colección de embeddings preentrenados y que utilizan el "enfoque de clasificación de verbos" basado en generar tokens **v** o **w** para verbos finitos con y sin sujeto explícito (respectivamente).

Se observa que si bien el enfoque de marcas de sujeto omitido (**0**) presentó un rendimiento inferior respecto a la medida F1, fue capaz de conseguir valores más altos de *accuracy*. Esto refleja un sesgo introducido por las características

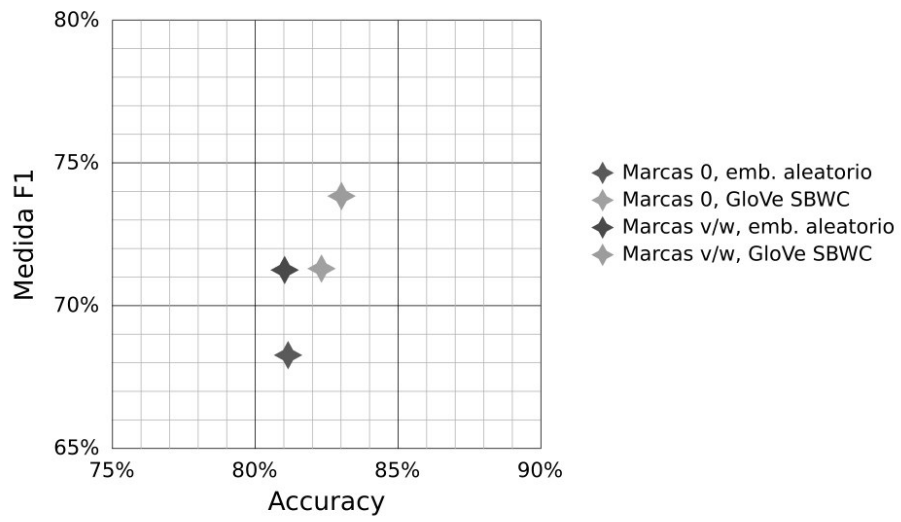

Figura 5.2: Comparación de las métricas *accuracy* y medida F1 para los modelos con dos capas ocultas entrenados para el experimento 1. Se observa que el enfoque basado en utilizar marcas **v** y **w** para verbos finitos con y sin verbo explícito presenta resultados superiores al enfoque de utilizar una marca **0** para indicar la posición de un sujeto omitido. Se aprecia además que utilizar embeddings preentrenados GloVe permite obtener mejores resultados en ambos casos.

del formato de salida: al requerirse la generación de un token adicional para marcar las oraciones con sujeto omitido, el modelo presenta una propensión hacia los resultados negativos (sin el token adicional) los cuales son más numerosos dentro de todas las particiones del corpus. Este sesgo hacia resultados negativos resulta en una mayor medida de *accuracy* al reducir el número de errores para la categoría mayoritaria (una menor cantidad de falsos positivos) pese a que se obtiene una mayor tasa de errores en la categoría minoritaria (las oraciones sin sujeto explícito, generando más falsos negativos). Esta propensión hacia la categoría negativa, sin embargo, lleva a una caída en la métrica de *recall* (la tasa de positivos reales recuperados, es decir, clasificados como verdaderos positivos), lo cual impacta en la medida F1, Los modelos con el enfoque de marcas de sujeto omitido no consiguieron obtener tasas de *recall* superiores al 70 %, inferiores a los valores de *recall* obtenidos con el enfoque de clasificación de verbos. Si bien el menor número de falsos positivos en el primer enfoque permitió que sus modelos mostraran una mejor precisión, esto no resulta suficiente para compensar el efecto de la métrica *recall* en el cómputo de la medida F1.

5.2.2. Evaluación de embeddings

Los experimentos iniciales mostraron que el uso de una colección de embeddings preentrenada (GloVe SBWC) permitía obtener un rendimiento superior respecto a un modelo análogo con embeddings aleatorios autogenerados. En la segunda instancia de experimentación se evaluó el uso de otras colecciones de embeddings: Word2Vec SBWC y emb39. Cada colección se probó en dos modelos con el el enfoque de clasificación de verbos y dos capas ocultas: uno con 500 unidades por capa y otro con 1000 unidades por capa. La tabla 5.2 presenta los resultados obtenidos para los modelos evaluados.

Embeddings	Capas ocultas	TER	Verbos no detectados	Accuracy	Precision	Recall	Medida F1
Aleatorio	2×500	4,70 %	4,68 %	81,04 %	69,06 %	73,61 %	71,26 %
Aleatorio	2×1000	4,89 %	**4,53 %**	82,61 %	74,27 %	**78,16 %**	71,09 %
Glove SBWC	2×500	4,68 %	5,92 %	83,03 %	73,06 %	74,67 %	73,85 %
Glove SBWC	2×1000	**4,64 %**	6,32 %	**83,25 %**	**77,87 %**	71,99 %	**74,81 %**
Word2Vec SBWC	2×500	5,12 %	7,64 %	80,42 %	75,19 %	65,77 %	70,17 %
Word2Vec SBWC	2×1000	4,78 %	6,20 %	81,74 %	76,61 %	64,82 %	70,22 %
emb39	2×500	5,03 %	7,02 %	80,42 %	73,37 %	66,63 %	69,84 %
emb39	2×1000	4,86 %	6,64 %	82,09 %	77,74 %	67,11 %	72,04 %

Tabla 5.2: Experimento 2: Mejores métricas logradas por cada modelo durante su entrenamiento, medidas respecto al corpus de desarrollo.

Como se puede observar en el diagrama 5.3, la incorporación de la colección de embeddings GloVe SBWC permite obtener mejores resultados que al utilizar embeddings construidos con la técnica Word2Vec. Se observó sin embargo que un modelo de 2×1000 unidades con la colección de embeddings emb39 fue capaz de obtener la menor tasa de error en traducción (TER) de todos los modelos entrenados, con un valor de 4,46 % (aunque dicho modelo mostró una medida F1 menor, de solo un 69,26 %), lo cual indica que los embeddings creados con la técnica Word2Vec podrían resultar más efectivos para otras aplicaciones de

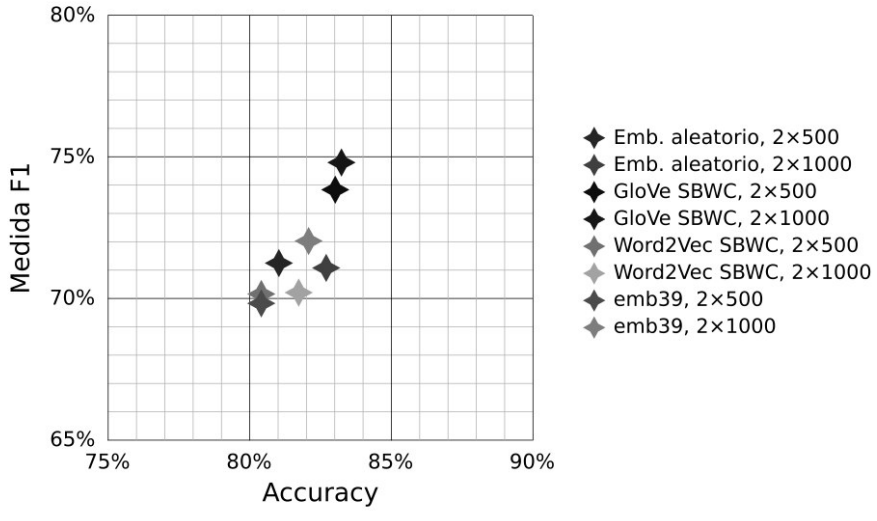

Figura 5.3: Comparación de las métricas *accuracy* y medida F1 para los modelos entrenados para el experimento 2. Se aprecia que los modelos entrenados con embeddings GloVe superan a los demás en ambas métricas.

PLN (en este caso, demostraron un mejor rendimiento para la tarea auxiliar de *POS-tagging*).

5.2.3. Exploración de número de capas ocultas y unidades por capa

Tras identificarse que la combinación del enfoque de marcas **v** y **w** y el uso de word embeddings GloVe permitía obtener un mayor rendimiento en redes de dos capas ocultas con 500 o 1000 unidades por cada, se procedió a realizar experimentos utilizando redes neuronales de otras dimensiones, como se muestra en la tabla 5.3. Las dimensiones consideradas se vieron limitadas por las características del hardware utilizado; no fue posible considerar modelos de más de 1500 unidades por capa (o de 1000 unidades en el caso de modelos con 4 capas) ya que los requerimientos para el entrenamiento del modelo excedían la memoria dedicada de la GPU utilizada, aun disminuyendo el tamaño de los *batches* utilizados por OpenNMT.

Como puede en el gráfico 5.4, aumentar las dimensiones del modelo (número de capas ocultas y cantidad de unidades por capa oculta) no necesariamente contribuye a la obtención de mejores resultados. Si bien aumentar el número de parámetros a entrenar tiene el potencial de ampliar la capacidad expresiva del modelo, un mayor número de parámetros también puede resultar en una mayor propensión a que la optimización del modelo se estanque en un máximo relativo no óptimo. Se observó que los modelos con 4 capas ocultas fueron incapaces de

Capas ocultas	TER	Verbos no detectados	Accuracy	Precision	Recall	Medida F1
2×500	4,68 %	5,92 %	83,03 %	73,06 %	**74,67 %**	73,85 %
2×1000	4,64 %	6,32 %	83,25 %	77,87 %	71,99 %	74,81 %
2×1500	5,48 %	6,47 %	83,55 %	**82,12 %**	68,07 %	74,44 %
3×500	**4,44 %**	6,69 %	82,66 %	76,51 %	72,56 %	74,48 %
3×1000	5,73 %	**5,70 %**	**84,15 %**	78,31 %	73,52 %	**75,84 %**
3×1500	5,56 %	6,02 %	83,55 %	81,39 %	67,30 %	73,68 %
4×500	30,93 %	14,78 %	66,91 %	56,42 %	43,69 %	49,25 %
4×1000	27,66 %	9,41 %	72,51 %	60,60 %	46,46 %	52,60 %

Tabla 5.3: Experimento 3: Exploración de redes de diferentes dimensiones, utilizando el enfoque (marcas **v** y **w**) y la colección de embeddings preentrenados (Glove SBWC) que presentaron un mayor rendimiento en los experimentos previos. Medido respecto al corpus de desarrollo.

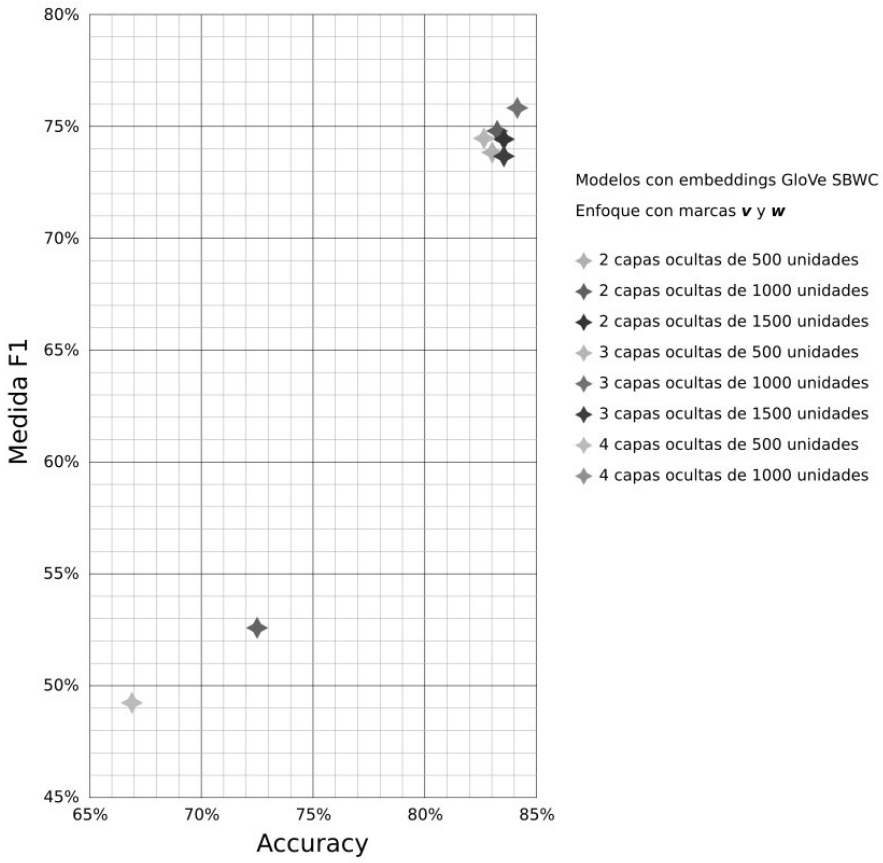

Figura 5.4: Comparación de las métricas *accuracy* y medida F1 para los modelos entrenados para el experimento 3.

obtener una performance similar a la de los modelos de 2 o 3 capas.

Los mejores resultados se obtuvieron con un modelo de 3 capas ocultas de 1000 unidades, siendo este el único de los modelos considerados capaz de obtener una medida F1 superior al 75 %. Se observa, sin embargo, que estos resultados son comparables a los de modelos de una menor dimensionalidad (con 2 capas de 500 o 1000 unidades), los cuales pueden entrenarse con menos recursos en cuanto a hardware y tiempo, como se muestra en la tabla 5.4.

Capas ocultas	Medida F1	Iteraciones antes de converger	Tiempo de entrenamiento
2×500	73,85 %	6000	1 hora
2×1000	74,81 %	12000	2,5 horas
3×1000	75,84 %	90000	10,5 horas

Tabla 5.4: Comparación del entrenamiento de modelos seleccionados

5.2.4. Evaluación con subcorpus de testing

Los resultados que se presentaron en las secciones precedentes se computaron utilizando la partición del corpus de "desarrollo" (*development*). Si bien el uso de una partición o subcorpus separado del utilizado para el entrenamiento contribuye a evitar que los resultados se vean sesgados por efectos espurios como *overfitting*, el uso de un mismo subcorpus para evaluar las diferentes configuraciones hiperparamétricas del modelo podría asimismo introducir sesgos: los modelos para los que se observó una mayor performance podrían explotar relaciones espurias dentro de la partición de desarrollo que no generalizan a ejemplos por fuera del mismo. Por este motivo, resulta necesario evaluar los modelos frente a un tercer conjunto de datos: la partición de evaluación o *testing*.

Capas ocultas	TER	Verbos no detectados	Accuracy	Precision	Recall	Medida F1
2×500	4,96 %	7,01 %	80,90 %	72,09 %	70,86 %	70,86 %
2×1000	**4,62 %**	6,44 %	82,63 %	76,79 %	69,62 %	73,03 %
3×1000	5,73 %	**5,09** %	**84,39 %**	**78,05 %**	**72,48 %**	**75,16 %**

Tabla 5.5: Evaluación de los tres modelos más prometedores (identificados en la subsección anterior) sobre la partición de testing.

Como se observa en la tabla 5.5, las métricas obtenidas al evaluar sobre el subcorpus de testing son comparables a las obtenidas sobre el subcorpus de desarrollo (tabla 5.3) por lo que se puede concluir que los resultados obtenidos generalizan a instancias por fuera del corpus. Se continúa observando un rendimiento superior al utilizar modelos de mayores dimensiones aunque estos presentan un costo de entrenamiento considerablemente mayor (ver tabla 5.4) por lo que en una aplicación práctica podría resultar más conveniente optar por un modelo que requiera menores recursos computacionales para su entrenamiento, aun al costo de una leve pérdida de performance. También se observa una

diferencia en los tiempos requeridos para procesar oraciones una vez entrenado el modelo (tabla 5.6), lo cual podría también llevar a que resulte preferible aceptar resultados de una calidad inferior si el tiempo de ejecución resultara un factor crítico.

Capas ocultas	Medida F1	Tiempo de procesamiento por entrada
2×500	70,86 %	10,20 ms
2×1000	73,03 %	16,22 ms
3×1000	75,16 %	19,20 ms

Tabla 5.6: Tiempo promedio insumido para procesar enunciados, medido en base a los 1623 enunciados del subcorpus de testing.

Los resultados obtenidos superan las marcas obtenidas por Ferrández en el primer antecedente encontrado para el problema (Ferrández y Peral, 2000), donde se reporta un accuracy de 75 %, pero el rendimiento de estos modelos no puede compararse directamente con otros estudios anteriores como Rello (2012), González (2018) y Gerez (2018) ya que los mismos abordan el problema de detección de sujeto omitido de un modo distinto, contemplando los verbos que aparecen en un uso impersonal (que no admite sujeto) como una tercera categoría. Por este motivo, se procedió a adaptar los modelos a un esquema de clasificación ternaria, como se explica en la siguiente sección.

5.3. Clasificación ternaria - detección de oraciones impersonales

Para adaptar los modelos propuestos en la sección anterior a un esquema de clasificación ternaria (con distinción entre oraciones personales e impersonales), se modificó el formato de salida utilizado en el entrenamiento a modo de contemplar tres posibilidades para un verbo finito:

- Verbo con sujeto explícito (etiqueta **v**) como “hace” en “Juan hace surf” (el sujeto, “Juan”, aparece explícitamente en el enunciado).
- Verbo con sujeto omitido (etiqueta **w**) como “hace” en “Hace un pastel” (el enunciado no indica quién hizo el pastel, pero queda implícita la existencia de un sujeto de tercera persona singular: “él” o “ella”).
- Verbo en un uso impersonal (etiqueta **y**), como “hace” en “Hace muchos años” (no se puede identificar un sujeto; “muchos años” opera solamente como complemento lo cual explica la ausencia de concordancia en número).

La clasificación de los verbos en el corpus para el entrenamiento del modelo se realizó a mediante una combinación del uso de atributos contemplados por AnCora y etiquetado manual mediante el procedimiento detallado en el anexo 3.

Empleando este conjunto de datos modificado se entrenaron modelos de dos capas ocultas de 500 unidades, dos capas ocultas de 1000 unidades y tres capas ocultas de 1000 unidades. Se observó que mientras que los modelos con dos capas ocultas requirieron menos de 20 000 iteraciones sobre el corpus de entrenamiento para converger (obteniendo una tasa de errores de traducción o TER de 4,61 % y 4,70 %), el modelo con tres capas ocultas no convergió hasta pasados los 190 000 "pasos" de entrenamiento, punto a partir del cual la medida TER comenzó a decaer, presumiblemente debido a sobreajuste u *overfitting*. Pese a esto, la menor tasa de errores alcanzada por el modelo de tres capas fue considerablemente más alta que en los modelos de dos capas (16,46 %) y los resultados del experimento sobre el subcorpus de desarrollo presentaron una calidad inferior como se muestra en la tabla 5.7. Una interpretación posible para este resultado es que el espacio paramétrico de mayor dimensionalidad que debe explorarse para el modelo de tres capas lleva a que su exploración durante el proceso de entrenamiento sea más propensa a verse atraída hacia un mínimo relativo más alejado al mínimo absoluto. La evolución de los modelos se presenta en las figuras 5.5 y 5.6.

Capas ocultas	Pasos de entrenamiento	Tiempo de entrenamiento	Menor valor de TER	Mayor valor macro F1
2×500	20 000, evaluado cada 2 000	1 hora	**4,61 %** (a los 18 000 pasos)	69,82 % (a los 10 000 pasos)
2×1000	20 000, evaluado cada 2 000	2,5 horas	4,70 % (a los 14 000 pasos)	**70,90 %** (a los 12 000 pasos)
3×1000	200 000, evaluado cada 10 000	16,5 horas	16,19 % (a los 190 000 pasos)	56,44 % (a los 110 000 pasos)

Tabla 5.7: Entrenamiento de modelos con datos para clasificación ternaria. Resultados evaluados sobre el subcorpus de desarrollo. Se presenta como medida "macro F1" al promedio de la medida F1 de cada categoría frente a las dos restantes.

Los modelos que obtuvieron un mayor rendimiento sobre el subcorpus de desarrollo (evaluado a través de la medida macro F1) fueron evaluados sobre la partición de testing obteniendo los resultados presentados en la tabla 5.8, obteniéndose nuevamente valores similares a los observados sobre desarrollo, lo cual permite descartar un sobreajuste a dicho subcorpus.

Capas ocultas	Entrenamiento	F1 sujeto explícito	F1 sujeto omitido	F1 impersonal
2×500	10 000 pasos	90,12 %	**76,85 %**	**50,47 %**
2×1000	12 000 pasos	**90,69 %**	76,29 %	46,43 %
3×1000	110 000 pasos	86,05 %	63,31 %	15,05 %
Capas ocultas	Macro F1	F1 ponderado	Accuracy	TER
2×500	**72,48 %**	86,19 %	82,96 %	4,93 %
2×1000	71,14 %	**86,41 %**	**84,14 %**	**4,92 %**
3×1000	54,80 %	79,27 %	76,84 %	16,31 %

Tabla 5.8: Medidas F1 para cada categoría en los experimentos de clasificación ternaria sobre el subcorpus de testing, promedio de medidas F1 (macro F1), macro F1 ponderado por el número de ejemplos de cada categoría (más influenciado por la categoría mayoritaria de sujetos explícitos), *accuracy* general (porcentaje de clasificaciones correctas) y tasa de error de traducción (TER) para el problema de POS-tagging y clasificación de verbos.

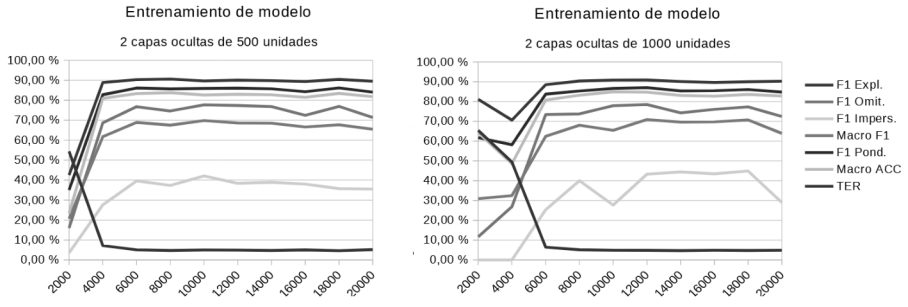

Figura 5.5: Evolución de métricas durante el entrenamiento de los modelos de 2 capas. En ambos casos, se observa que el *fitness* del modelo, identificable como la tasa de error en traducción (TER), tiende a estabilizarse tras un número limitado de iteraciones sobre el corpus de entrenamiento (4000 y 6000 en los modelos con capas de 500 y 1000 unidades, respectivamente) aunque esta métrica continúa mejorando a un ritmo menor (inferior a 1 % cada 2000 pasos) hasta encontrarse un valor óptimo a los 18 000 y 14 000 pasos, tras los cuales el modelo presentó un TER mayor atribuible a sobreajuste. La inestabilidad en la evolución de las métricas de clasificación se debe a que el problema de optimización de OpenNMT (minimizar el TER para al transformar la oración a POS-tagging con las marcas especiales definidas) no depende directamente del problema de clasificación de verbos por lo que un modelo puede continuar optimizándose respecto a su tasa de errores que refleja un POS-tagging más correcto pero resultando en más clasificaciones erróneas de verbos.

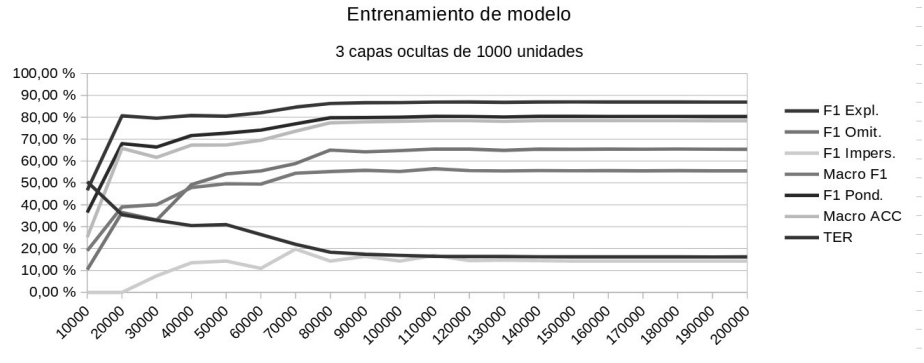

Figura 5.6: Evolución de métricas durante el entrenamiento del modelo de 3 capas. El modelo presentó una convergencia notoriamente más lenta que los modelos de 2 capas, requiriendo 200 000 iteraciones sobre el corpus de entrenamiento. Si bien el descenso de la tasa de error de traducción (TER) mostró una desaceleración tras los primeros 100 000 pasos de entrenamiento (tras lo cual se estabilizan también las demás métricas), continuó descendiendo hasta alcanzar los 16,19 % a los 190 000 pasos, tras lo cual se observó un aumento a 16,20 % en el paso 200 000. Pese al costo computacional significativamente mayor del entrenamiento, el modelo de 3 capas presentó un rendimiento inferior en todas las métricas.

Se aprecia un rendimiento comparable para los dos modelos de dos capas mientras que el modelo de tres capas continúa presentando un rendimiento inferior en todas las métricas. Todos los modelos muestran un mayor rendimiento en la detección de verbos con sujeto explícito y un rendimiento más pobre en la detección de verbos impersonales, lo cual coincide con la proporción de ejemplos en el corpus, donde un 73 % de los verbos en la partición de entrenamiento presentan sujeto explícito y solo un 1,5 % corresponden a verbos impersonales.

5.3.1. Comparación con clasificadores binarios

Los resultados de los modelos de clasificación ternaria (sujeto explícito vs omitido vs impersonal) pueden compararse con los los obtenidos con los clasificadores binarios (sujeto explícito vs sin sujeto explícito) de la sección anterior agrupando los resultados de “sujeto omitido” y “verbo impersonal”. Como se muestra en la tabla 5.9, los resultados de F1 “binario” obtenidos por los clasificadores ternarios con dos capas ocultas superan a los obtenidos por los modelos binarios de iguales dimensiones. Bajo esta métrica, el modelo que presentó la mejor performance fue el modelo de clasificación ternario de dos capas ocultas de 500 unidades.

Capas ocultas	F1 en modelo binario	F1 en modelo ternario
2×500	71,47 %	**77,85 %**
2×1000	73,03 %	76,88 %
3×1000	**75,16 %**	63,94 %

Tabla 5.9: Medidas F1 para la detección de ausencia de sujeto (por sujeto omito o por uso impersonal del verbo). Resultados evaluados sobre el subcorpus de testing.

5.3.2. Comparación con antecedentes

En la tabla 5.10 se comparan los resultados obtenidos en el presente estudio con los observados en los estudios listados como antecedentes (Ferrández y Peral, 2000) (Rello y cols., 2012) (González y Martínez, 2018) (Gerez y cols., 2019). Debe tenerse en cuenta que las distintas aproximaciones al problema emplearon múltiples enfoques sobre distintos corpus y que aun en los casos en los que se trabajó sobre un mismo corpus, los criterios para la clasificación de cada verbo pueden ser distintos. En particular, se nota que en González y Martínez (2018) se utilizó también el corpus AnCora, aunque únicamente se consideraron como impersonales los verbos en oraciones reflexivas como "(se) apoya" en *"se apoya iniciativas"* pero no otras construcciones consideradas como impersonales en este estudio como "hace" en *"hace cuatro años"*.

Modelo	Corpus	F1 binario	F1 Expl.	F1 Omit.	F1 Impers.	Macro F1	Accuracy
2×500 binario	AnCora	71,47 %	-	-	-	-	80,90 %
2×1000 binario	AnCora	73,03 %	-	-	-	-	82,63 %
3×1000 binario	AnCora	75,16 %	-	-	-	-	84,39 %
2×500 ternario	AnCora	**77,85 %**	90,12 %	**76,85 %**	50,47 %	72,48 %	82,96 %
2×1000 ternario	AnCora	76,88 %	90,69 %	76,29 %	46,43 %	71,14 %	84,14 %
3×1000 ternario	AnCora	63,94 %	86,05 %	63,31 %	15,05 %	54,80 %	76,84 %
González y Martínez (2018)	AnCora	72,8 %	**90,9 %**	71,8 %	**55,5 %**	**72,7 %**	**86,1 %**
Gerez et al. (2019)	AnCora	71 %	89 %	67 %	34 %	63 %	82 %
Ferrández (2000)	LexEsp	**92,67 %**	-	-	-	-	**90,40 %**
Ferrández (2000)	Blue Book	78,30 %	-	-	-	-	86,00 %
Rello et al. (2012)	ESZIC	77,4 %	**91,2 %**	75,5 %	**72,7 %**	**79,8 %**	86,7 %
González y Martínez (2018)	ESZIC	69,5 %	89,1 %	66,3 %	62,2 %	73,9 %	83,3 %

Tabla 5.10: Comparación entre los resultados obtenidos en este estudio y en estudios antecentes. Se remarca en negrita los mejores resultados para cada métrica utilizando el corpus AnCora (directamente comparables a este estudio) y en general.

A nivel del problema de clasificación binario sobre la presencia o ausencia de un sujeto explícito (considerando el caso de usos impersonales como idéntico al sujeto omitido), el estudio de Ferrández y Peral (Ferrández y Peral, 2000) continúa presentando el mejor resultado dentro de la literatura disponible aunque, como se menciona en la sección de Antecedentes, los corpus utilizados en dicho estudio tienen particularidades (dimensión muy reducida, distribución de ejemplos con mayor proporción de sujetos implícitos que otros corpus) lleva a que una comparación directa entre este estudio y estudios precedentes pueda

resultar inadecuada, como se plantea en (Rello y cols., 2012). Por fuera de dicho estudio, la mejor medida F1 para el problema de clasificación binaria fue la que se observó para el modelo de clasificación ternaria de dos capas de 500 unidades ocultas propuesto en este trabajo (F1 de 0.7785), el cual presenta también el mejor resultado en cuanto a detección de sujeto omitido en el escenario de clasificación ternaria (no contemplado por Ferrández), obteniéndose una medida F1 de 0.7685 frente a la medida de 0.755 obtenida por el estudio Elliphant (Rello y cols., 2012), sobre el corpus ESZIC, que constituye el estado del arte hasta el momento así como la métrica de 0.718 obtenida por González y Martínez (2018) sobre el corpus AnCora.

Por el contrario, los modelos propuestos en este trabajo no lograron presentar un rendimiento de estado del arte con respecto a la identificación de verbos en usos impersonales, lo cual pudo deberse a que la proporción de verbos impersonales dentro del corpus AnCora es muy reducida (en el entorno de 1,5 %), un problema que también afectó al estudio de Gerez, Irazusta e Irace, antecedente que también utilizó redes neuronales entrenadas sobre el mismo corpus. (Gerez y cols., 2019)

Los resultados obtenidos por el modelo de mayor rendimiento (clasificador ternario de dos capas de 500 unidades) presentan la mayor medida F1 en cuanto a clasificación binaria sobre el corpus AnCora (0.7785, frente a 0.728 por Gonzalez y Martínez) mientras que en clasificación ternaria se obtuvieron resultados de estado del arte tanto en la medida F1 para verbos con sujeto omitido (0.7685).

Verbos no detectados y clasificaciones espurias

Debido al uso de un modelo secuencia-a-secuencia en este proyecto, los modelos propuestos en este estudio presentan una desventaja no encontrada en los clasificadores propuestos en trabajos anteriores: la posibilidad de que un verbo no se detecte en el proceso de "traducción" de OpenNMT y, por ende, no se obtenga un valor de clasificación. Esta posibilidad está contemplada en las figuras para métricas F1, donde la no-detección de un verbo se considera como un falso negativo para la clase correspondiente, disminuyendo el factor *recall*. La tabla 5.11 presenta estadísticas sobre los verbos no detectados en la evaluación final de los modelos propuestos; se aprecia que los verbos con sujeto omitido son más propensos a esta clasificación errónea, presumiblemente por poder aparecer en oraciones que presentan una sintaxis menos habitual que el orden sujeto-verbo-objeto típico del castellano.

Modelo	Número de verbos finitos	Verbos no detectados	Suj. expl.	Suj. omitido	Impersonales
2×500	4162	205 (7,47 %)	**106 (6,74 %)**	101 (9,98 %)	0 (0 %)
2×1000	4162	**226 (5,43 %)**	145 (4,77 %)	**81 (7,63 %)**	**0 (0 %)**
3×1000	4162	289 (6,94 %)	185 (6,08 %)	101 (9,51 %)	3 (5,26 %)

Tabla 5.11: Verbos no detectados para los modelos ternarios; evaluado sobre el subcorpus de testing.

De un modo similar, los modelos presentan la posibilidad de que la salida

presente verbos “espurios”, valores de clasificación que no corresponden a ningún verbo finito en la entrada. Se tomó el criterio de no considerar este fenómeno para el cálculo de métricas ya los valores sobrantes pueden descartarse fácilmente en un posprocesamiento de la salida. Como se observa en la tabla 5.12, la tasa de incidencia de clasificaciones espurias es baja, en el entorno de un 1 % del número de instancias reales.

Modelo	Número de verbos finitos	Valores espurios	Suj. expl.	Suj. omitido	Impersonales
2×500	4162	**13 (0,31 %)**	**7 (0,23 %)**	**6 (0,56 %)**	**0 (0 %)**
2×1000	4162	43 (1,03 %)	27 (0,89 %)	16 (1,51 %)	**0 (0 %)**
3×1000	4162	124 (2,98 %)	88 (2,89 %)	35 (3,30 %)	1 (1,75 %)

Tabla 5.12: Valores de clasificación espurios para los modelos ternarios; evaluado sobre el subcorpus de testing.

5.4. Análisis de errores

Entre los modelos propuestos para este proyecto, los clasificadores “ternarios” basados en redes neuronales de dos capas ocultas mostraron el mejor rendimiento, obteniéndose la mayor medida F1 para reconocimiento de sujetos omitidos (0.7685 sobre el subcorpus de testing) para un modelo con 500 unidades por capa oculta y el mejor valor de *accuracy* (84,14 %) para el modelo de 1000 unidades por capa oculta. En la presente sección, se examinarán los resultados obtenidos por dichos modelos con el objetivo de detectar ante qué tipo de escenarios los clasificadores resultan más propensos a fallar.

5.4.1. Matrices de confusión y errores entre categorías

En las tablas 5.13 y 5.13 se presentan las matrices de confusión para los modelos considerados, las cuales se presentan gráficamente en los diagramas de Sankey de las figuras 5.7 y 5.8.

Categoría real	Detectado como explícito	Detectado como implícito	Detectado como impersonal	No detectado
Sujeto explícito	**2636 (86,63 %)**	187 (6,15 %)	15 (0,49 %)	205 (6,74 %)
Sujeto omitido	158 (14,88 %)	**790 (74,39 %)**	8 (0,75 %)	106 (9,98 %)
Verbo impersonal	13 (22,81 %)	17 (29,82 %)	**27 (47,37 %)**	0 (0 %)
(Espurio)	7	6	0	-

Tabla 5.13: Matriz de confusión para clasificador ternario de dos capas ocultas de 500 unidades, evaluado sobre partición de testing.

Puede observarse que en las clasificaciones erróneas más frecuentes para instancias con sujeto explícito o implícito corresponden o bien a la no detección de verbos durante el proceso de traducción (problema con una incidencia sensiblemente mayor en el modelo de 500 unidades por capa, el cual falla en la detección de un 7,47 % de los verbos frente a 5,43 % en el modelo de 1000 unidades por capa) o a la confusión entre estas dos categorías. Este comportamiento se ilustra en la figura 5.9.

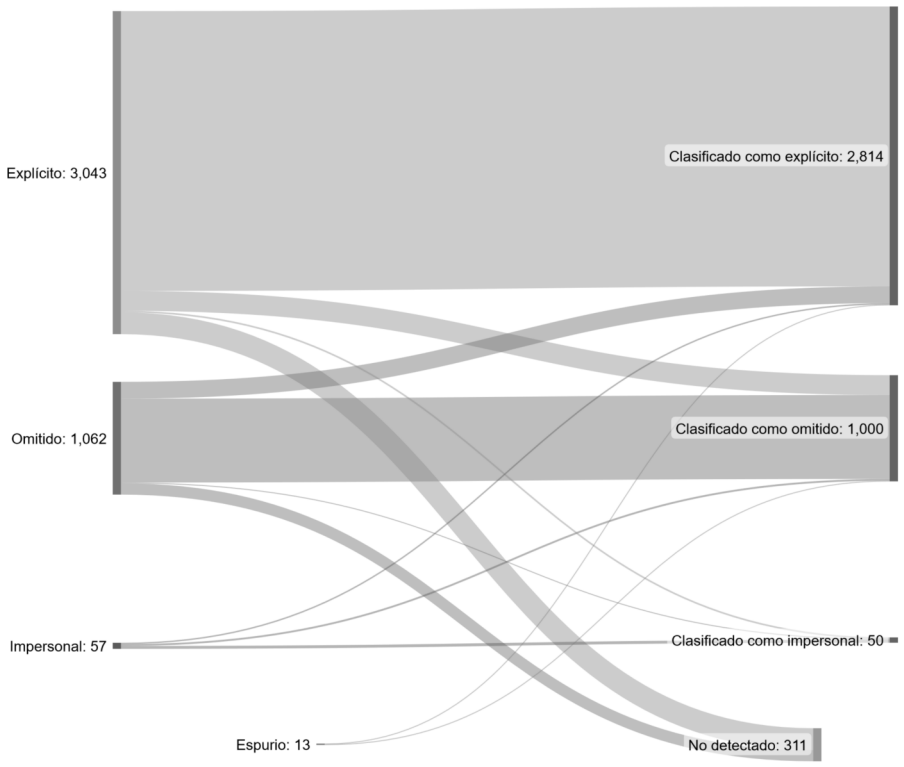

Figura 5.7: Representación gráfica como diagrama de Sankey de la matriz de confusión del clasificador de dos capas ocultas de 500 unidades.

Categoría real	Detectado como explícito	Detectado como implícito	Detectado como impersonal	No detectado
Sujeto explícito	**2707 (88,96 %)**	173 (5,69 %)	18 (0,59 %)	145 (4,77 %)
Sujeto omitido	201 (18,93 %)	**769 (72,41 %)**	11 (1,04 %)	81 (7,63 %)
Verbo impersonal	19 (33,33 %)	12 (21,05 %)	**26 (45,61 %)**	0 (0 %)
(Espurio)	27	16	0	-

Tabla 5.14: Matriz de confusión para clasificador ternario de dos capas ocultas de 1000 unidades, evaluado sobre partición de testing.

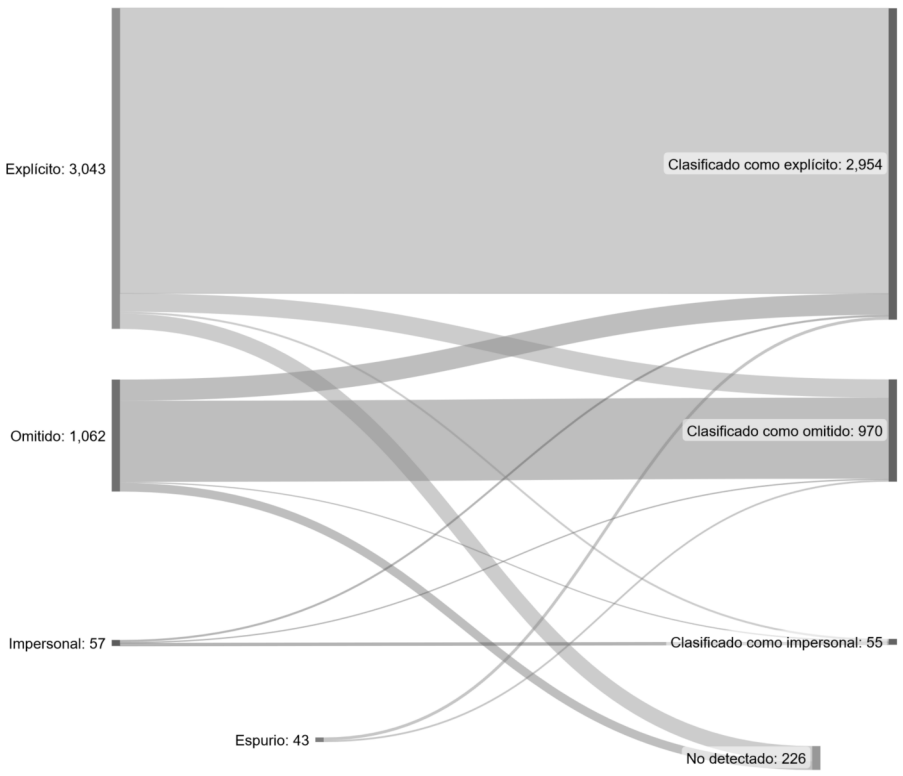

Figura 5.8: Representación gráfica como diagrama de Sankey de la matriz de confusión del clasificador de dos capas ocultas de 1000 unidades.

La detección errónea de un verbo como impersonal es considerablemente menos frecuente, lo cual es atribuible a la mucho menor proporción de ejemplos etiquetados como "impersonal" en el corpus de entrenamiento, aunque esto lleva también a que la tasa de verbos impersonales correctamente clasificados sea menor, siendo la única categoría en la que los clasificadores propuestos fueron incapaces de generar la etiqueta correcta para la mayoría de las instancias (solo lográndose en el 47,37 % y el 45,61 % de los casos para modelos de 500 y 1000 unidades por capa).

Como se muestra en la figura 5.9, la distribución de las instancias de verbo impersonal clasificadas incorrectamente varía entre modelos, observándose una predominancia de las clasificaciones como sujeto omitido (coherente con la inexistencia de un sujeto en la frase impersonal) para el modelo de 500 unidades por capa mientras que en el modelo mayor resulta más común la clasificación en la clase de sujeto explícito (presumiblemente efecto de un sesgo al ser la clase mayoritaria en el corpus), sin encontrarse en ningún caso verbos impersonales no detectados por el modelo. Sin embargo, dado el reducido número de ejemplos que componen estos resultados (solo 30 y 31 verbos impersonales de un total de 57 en el subcorpus de testing), estos resultados y su variación podrían no ser significativos.

5.4.2. Efecto de términos fuera de vocabulario

Durante la generación del modelo en OpenNMT, el framework construye un vocabulario de entrada y salida con los tokens encontrados en el corpus indicado para el entrenamiento y validación interna del modelo (en este caso, las particiones de entrenamiento y desarrollo del corpus AnCora). Como resultado se obtuvo un vocabulario de 38 714 palabras para la entrada (posteriormente mapeadas a vectores de 300 dimensiones con la colección embeddings preentrenados) y de 15 etiquetas (POS y marcadores de las categorías de verbos) para la salida. Dado que la generación del vocabulario no contempló los valores en la partición de testing, las instancias en la evaluación pueden contener palabras por fuera del vocabulario (OOV, *out of vocabulary*), a las cuales se les asignará un mismo vector de embedding aleatorio de "token desconocido" (transcrito como $<unk>$) para el cual los modelos desconocerán *a priori* su clase gramatical. Debe tenerse en cuenta que los modelos tratan a estas palabras como valores numéricos que no reflejan su morfología por lo que diferentes inflexiones de un mismo lexema (variantes de género y número de un sustantivo o adjetivo, o formas conjugadas de un verbo) no se reconocerán como símiles ni se podrá realizar deducciones a partir de terminaciones (como "-aron" para verbos en pasado de la tercera persona plural) que a un hablante le permitirían deducir la clase léxica de un término no familiar.

Una exploración informal de los resultados del modelo sugiere que la incapacidad del modelo de reconocer la clase léxica de palabras por fuera del vocabulario es una importante causal de errores de no detección de verbos. Esto puede observarse en el ejemplo:

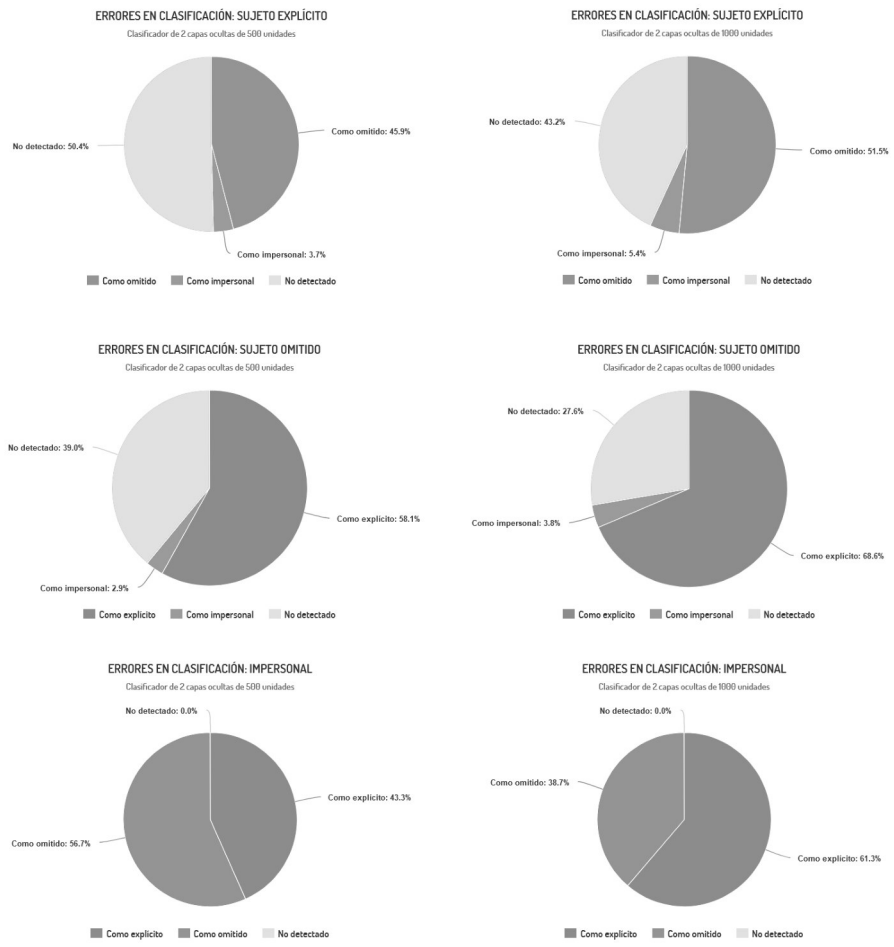

Figura 5.9: Clasificación de los errores encontrados para las tres categorías (sujeto explícito, omitido o uso impersonal) en los modelos de 2 capas ocultas con 500 unidades (izquierda) y 1000 unidades (derecha).

- *José Ramón Guimaraens ha negado que **prestase** a Jesús Gil 1.300 millones de pesetas en junio de 1992 para que el club rojiblanco **pudiese** transformarse en sociedad anónima.*

La oración presenta tres verbos finitos: "ha" ("ha negado", con sujeto explícito, Guimaraens), "prestase" (sujeto omitido; se deduce contextualmente que también refiere a Guimaraens) y "pudiese" (sujeto explícito, "el club rojiblanco") aunque el modelo de 2 capas de 1000 unidades solamente fue capaz de reconocer al primero mientras que las formas subjuntivas "prestase" y "pudiese", no encontradas en el vocabulario del modelo, fueron etiquetadas erróneamente como sustantivos. Si bien la categorización de elementos OOV como sustantivos no es universal (los modelos frecuentemente logran identificar correctamente verbos fuera del vocabulario), resulta natural asumir un sesgo hacia etiquetar casos dudosos como sustantivos (etiqueta **n**) al ser esta la categoría más frecuente dentro del corpus de entrenamiento (donde se encuentran 96 925 sustantivos frente a solo 13 489 verbos).

Asimismo, se encuentran casos en los que elementos OOV no verbales se reconocen como verbos, llevando a la generación de "verbos espurios" (valores de clasificación que no corresponden a un verbo finito presente en la entrada). Un ejemplo de esto es la oración:

- *Quizá porque las reses **bravas** se reconocen entre ellas.*

En este caso, el token en negrita "bravas" fue interpretado erróneamente como un verbo con sujeto explícito. Esto evidencia el problema de que las distintas formas inflecionales de un lexema se consideren como elementos distintos a efectos del vocabulario manejado por OpenNMT: el vocabulario de entrada incluye el elemento "bravo", pero no su forma femenina plural "bravas", lo que llevó a que "las reses bravas" se identificara como una frase con sujeto "las reses" seguido por un verbo en vez de como un grupo nominal con un adjetivo. Asimismo, al no considerar la morfología inflecional el modelo pudo asumir que "bravas" podría corresponder a un verbo con sujeto plural, mientras que un hablante nativo de español reconocería que la terminación "-as" en todo caso solo podría corresponder a una forma singular de un hipotético verbo **bravar*.

Esta observación sugiere que el rendimiento del clasificador podría incrementarse si la entrada tuviera un preprocesamiento de lematización (reemplazar o aumentar los elementos que el modelo toma como entrada con la forma canónica del lexema, como el infinitivo en caso de verbos) con el objetivo de que el modelo pueda reconocer un mayor número de formas verbales. En forma similar, las consideraciones de concordancia gramatical podrían incorporarse si la entrada ya hubiera pasado por una segmentación en morfemas (de modo que una forma verbal como "prestase" se procese como dos tokens: la raíz "prest-" que la identifica como una forma conjugada de "prestar" y la declinación "-ase" que la marca como una conjugación finita de tercera persona singular) aunque dicho etiquetado representa una tarea poco trivial.

Otra posible estrategia de mitigación es la empleada en Chiruzzo (2012), donde se optó por reemplazar los elementos fuera de vocabulario por los tokens

más comunes en el corpus perteneciente a la misma categoría léxica. Esto presupone realizar un reconocimiento *POS* como preprocesamiento de la entrada, lo cual constituye un problema ampliamente estudiado en la literatura con una elevada tasa de éxito. (Chiruzzo y Wonsever, 2020)

La potencial mitigación de este problema reviste interés particularmente en vista de que la no detección de verbos finitos representa un porcentaje muy significativo de los errores de clasificación, como lo ilustra la figura 5.9.

5.4.3. Errores en identificación de verbos impersonales

Los verbos impersonales considerados en este estudio incluyen, entre otros, los que aparecen en oraciones "impersonales reflejas" (Real Academia Española, 2011) donde un verbo en tercera persona singular es precedido por la partícula "se" (como "alerta" en "Se alerta del peligro"). Este tipo de construcciones constituye una mayoría entre los verbos impersonales del corpus, por lo que los errores en su tratamiento impactan muy significativamente en las métricas asociadas a la categoría impersonal.

El reconocimiento oraciones impersonales reflejas presenta un desafío dado que su estructura puede resultar idéntica a la de las oraciones "pasivas reflejas", oraciones personales donde el pronombre "se" hace referencia a un sujeto. A modo de ejemplo, considérense las siguientes oraciones cuya estructura resulta análoga:

- (A) *En este taller, se trabaja la madera.*
- (B) *En este congreso, se trabaja la problemática.*

Un análisis plausible es considerar que "se trabaja" en (A) forma parte de una oración impersonal refleja, donde "la madera" opera como complemento directo de una acción genérica mientras que resulta más razonable analizar que en (B) "se trabaja" (se trata) aparece en una oración pasiva refleja, donde "el tema" opera como sujeto. Dicho análisis sería respaldado si al reemplazar el argumento singular por un plural se encontrase que (A) mantiene el verbo en singular mientras que (B) lo cambia a plural como en:

- (A') *En este taller, se trabaja la madera, el metal y el mármol.*
- (B') *En este congreso, se trabajan la problemática la discriminación y su efecto en la economía.*

Dado que las oraciones pasivas reflejas poseen un sujeto identificable, no corresponde etiquetarlas como impersonales sino como oraciones personales que, según el caso, presentarán sujeto explícito u omitido. Si bien el corpus de entrenamiento y, por tanto, los modelos entrenados en este estudio siguen dicho criterio, la similitud sintáctica lleva a que un elevado número de oraciones impersonales reflejas se etiqueten como "con sujeto explícito", como sería el caso si se tratase de una oración pasiva refleja:

- *Por el momento no **se sabe** cuándo marchará el jefe de la delegación siria..*
- *Aunque **se justificó** la discreción en necesidades de la propia investigación..*

Asimismo, se encuentran casos en los que se da una clasificación errónea de un verbo etiquetado como impersonal según el criterio del corpus pero reconocido erróneamente como "con sujeto omitido" por el clasificador:

- *..no **se puede descartar** a toda una doble campeona olímpica...*

De forma similar, varias de las clasificaciones de verbos personales (ya sea con sujeto explícito u omitido) como "impersonales" responde a la confusión de oraciones pasivas reflejas con impersonales reflejas:

- *... el australiano ... con saques a 216 kilómetros por hora **se hizo** con el primer set.*
- *Creemos que si, como **ha ocurrido** en Chile, los ciudadanos...*

5.4.4. No reconocimiento de sujetos explícitos

Al igual que en trabajos anteriores, se identificó que los clasificadores presentaron dificultades al reconocer el sujeto explícito cuando el mismo aparece pospuesto al verbo en vez de antepuesto al verbo como marca el orden más típico sujeto-verbo-objeto de las oraciones en español. En estos casos el clasificador etiqueta la instancia equivocadamente como "con sujeto omitido". Entre las instancias afectadas se encuentra:

- *... porque no **vale** la revancha ...*
- *... se **grabó** una serie de televisión ...*
- *... no **ha sido** posible contar con todos ...*
- *También **han recibido** una misiva similar todas las ciudades candidatas ...*

Un fenómeno similar se aprecia en oraciones donde el sujeto precede al verbo pero no directamente sino que hay otra frase (por ejemplo una acotación) de por medio. Un ejemplo de esto es:

- *El suizo François Carrara, director general del COI, **ironizó** ayer, en Lausana, sobre ...*

En este caso el verbo "ironizó" fue etiquetado como "con sujeto omitido" pese a que puede identificarse a "François Carrara", nombrado antes en la frase, como su sujeto. Este comportamiento resulta esperable en vista de que las comas

(tratadas como un token más a efectos del modelo) a menudo separan grupos verbales en conjunciones donde, según el criterio de los creadores de AnCora, se considera que un sujeto "compartido" (a nivel semántico) solo aplica para el primer verbo de la conjunción. Este criterio dictamina que en una frase como "María corre, nada y conduce" se interprete que "nada" y "conduce" presentan sujeto omitido (como si la estructura subyacente de la frase fuera "María corre, (ella) nada y (ella) conduce"), llevando a que un verbo precedido por una coma tienda a interpretarse como de sujeto omitido.

Un tercer escenario en el cual resulta recurrente la clasificación errónea de sujetos explícitos es tras la palabra "que", la cual puede operar tanto como un pronombre que introduce una oración relativa ('el libro *que* estaba sobre la mesa", "el libro *que* estaba leyendo") o como una conjunción para introducir una cláusula subordinada ("te comenté *que* estaba leyendo ese libro"). Únicamente en el primer caso, el pronombre "que" puede operar como sujeto del verbo en la oración relativa, aunque únicamente si el referente externo a la oración relativa se corresponde con el sujeto del verbo interno a la oración relativa. Esto implica que de los tres ejemplos con el verbo "estaba" precedido de "que" solo corresponda identificar sujeto explícito en el primero:

- *El libro que* ***estaba*** *sobre la mesa*: Sujeto explícito (pronombre "que", referenciando al libro, cumple el rol sintáctico de sujeto)
- *El libro que* ***estaba*** *leyendo*: Sujeto omitido (que *yo* estaba leyendo; el pronombre "que" cumple un rol sintáctico de complemento respecto a "estaba leyendo")
- *Te comenté que* ***estaba*** *leyendo ese libro*: Sujeto omitido (que *yo* estaba leyendo; "que" aparece como conjunción, no es argumento de "estaba leyendo" sino que sirve para nominalizar la frase)

Es posible que la incidencia de casos como estos pueda mitigarse mediante la incorporación al corpus de entrenamiento de una mayor cantidad de instancias que presenten este tipo de estructuras, por ejemplo, al aumentar artificialmente el número de oraciones con sujeto pospuesto a fines de reducir el sesgo de que el sujeto en el español tiende a ubicarse antes del verbo; aunque esto podría aumentar el riesgo de que se reconozca erróneamente como sujeto al complemento que sigue a un verbo con sujeto omitido.

5.4.5. Efecto del largo de la entrada

Tradicionalmente los modelos secuencia a secuencia han presentado dificultades para tratar adecuadamente entradas con un número elevado de elementos dado que la influencia de los elementos codificados en los primeros tokens de la secuencia tendía a decaer exponencialmente a medida que se procesaban elementos sucesivos. La incorporación de mecanismos de atención (contemplados en los modelos de OpenNMT empleados en este proyecto) logran atenuar significativamente este problema aunque no lo resuelven por completo.

En el caso particular de este estudio, además de estas características estructurales de los modelos secuencia-a-secuencia, las entradas de mayor largo presentan más oportunidades para que se den escenarios que tienden a confundir al clasificador como los identificados en las subsecciones anteriores, como ser la inclusión de acotaciones parentéticas que separan al verbo de su sujeto o el uso de términos por fuera del vocabulario reconocido por el modelo.

Categoría real	Detectado como explícito	Detectado como implícito	Detectado como impersonal	No detectado
Sujeto explícito	**104,76**	126,87	144,56	109,77
Sujeto omitido	117,28	**104,45**	130,27	121,47
Verbo impersonal	110,74	123,17	**106,92**	(sin instancias)

Tabla 5.15: Largo promedio de la entrada (número de tokens) para cada combinación de clasificación real frente a clasificación generada por el modelo de dos capas y 1000 unidades por capa; se presenta los valores correspondientes a clasificaciones correctas en negrita. El largo promedio de las instancias en el subcorpus de testing es de 107,18.

Para analizar la incidencia de estos efectos, se procedió a analizar el largo promedio de la entrada para cada combinación de valor real y valor predicho por el modelo, resultando en la tabla 5.15. Se encontró que, para todas las categorías, las clasificaciones correctas consistentemente tendieron a producirse en ejemplos promedialmente más cortos (104 o 105 tokens) que la media general (107,18 tokens) mientras que, por el contrario, entre las clasificaciones incorrectas y no-detecciones el número de entradas más largas que la media se encuentra sobrerrepresentado.

En la figura 5.10 se aprecia que la proporción de clasificaciones correctas decrece muy significativamente para entradas que superan los 200 tokens de largo. Debe considerarse, empero, que las entradas con este largo representan solo un 4,5 % del total, por lo que la incidencia de este efecto, aunque caracterizable, resulta limitada en el total de clasificaciones.

Una posible estrategia de mitigación para este problema sería aplicar un segmentador de cláusulas verbales como preprocesamiento para la entrada, de manera que un enunciado largo compuesto por múltiples frases verbales se separe en segmentos más cortos para los que el clasificador se mostrará más efectivo.

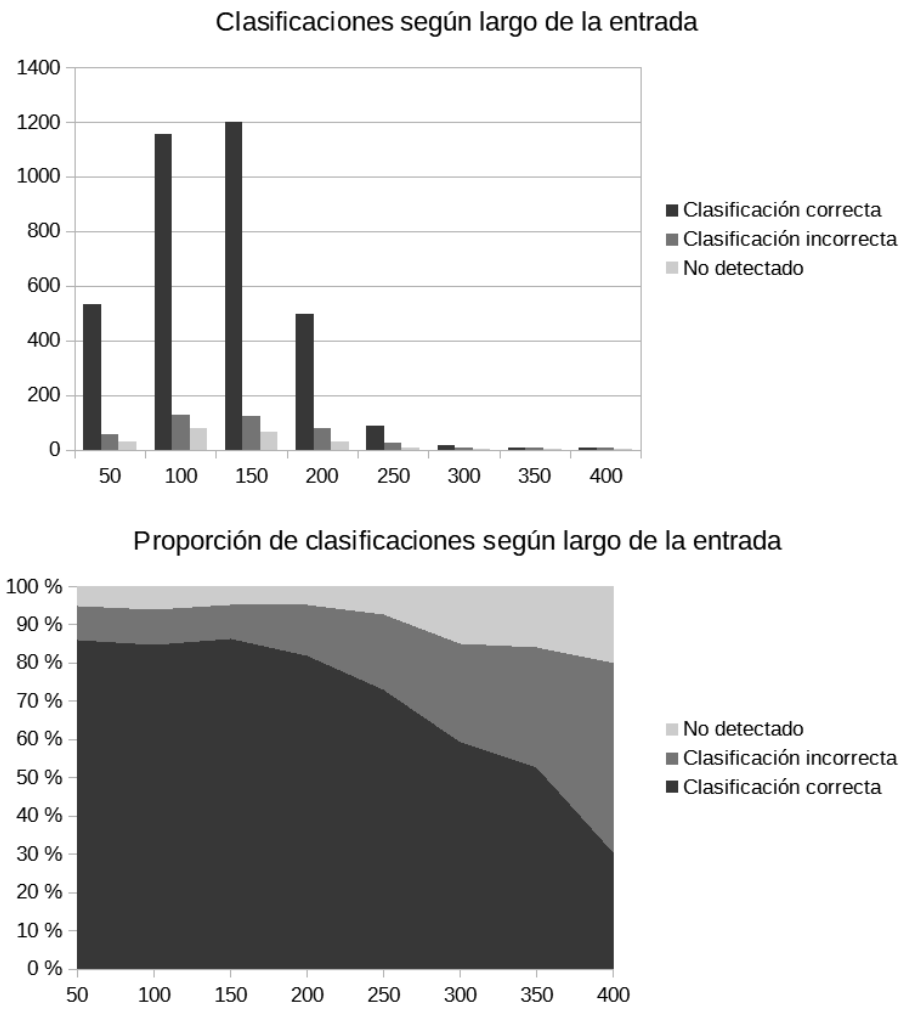

Figura 5.10: Resultados del modelo de dos capas ocultas de 1000 unidades sobre instancias del subcorpus de testing según el largo (número de tokens) de las mismas, agrupado por rangos de 50.

Capítulo 6

Conclusiones y trabajo futuro

6.1. Conclusiones

En el transcurso de este proyecto se transformó y amplió el corpus AnCora en español para la generación de corpus paralelos orientados a la detección del fenómeno de sujeto omitido y de oraciones impersonales a través de un proceso de traducción automática. Se exploraron tres formatos distintos basados en etiquetas de clase gramatical (*POS*) para reconocer esta característica: uno en el que se empleó una etiqueta especial **0** para representar la posición de un sujeto omitido (siguiendo el criterio de AnCora), otro en el que la distinción entre verbos con y sin sujeto explícito se realizó a través de etiquetas diferentes para uno y otro caso (**v** para verbos finitos con sujeto explícito, **w** en el caso contrario) y, finalmente, una extensión del segundo formato para distinguir entre verbos personales con sujeto omitido (etiqueta **w**) y verbos en un uso impersonal (etiqueta **y**).

A partir de estos corpus transformados se utilizó el framework de traducción automática OpenNMT para entrenar múltiples modelos de traducción cuya salida se postprocesó para construir algoritmos de clasificación entre las categorías verbales consideradas (verbos con sujeto explícito, con sujeto omitido y, en el caso del tercer formato, verbos impersonales) con un elevado nivel de efectividad. Se exploraron diversas configuraciones hiperparamétricas para determinar qué modelos presentaban un mayor rendimiento lo cual permitió identificar los clasificadores basados en formatos y configuraciones más prometedoras, evaluar su rendimiento ante un subcorpus reservado para testing y comparar los resultados con estudios precedentes.

Los resultados experimentales obtenidos demostraron que los clasificadores ternarios (con etiquetas **v**, **w** e **y** para verbos con sujeto explícito, omitido e impersonales) basados en modelos de OpenNMT con dos capas ocultas de 500 y 1000 unidades por capa y empleando embeddings pre-entrenados en forma-

to GloVe fueron capaces de resolver el problema planteado con un rendimiento comparable al obtenido en los antecedentes del estudio y, en determinadas métricas, alcanzando valores que representan un avance para el estado del arte. En particular, uno de los modelos propuestos fue capaz de obtener una métrica F1 de 0,7685 para la identificación de verbos con sujeto omitido (en oposición a sujetos explícitos o verbos en un uso impersonal), lo cual constituye el valor más alto para esta métrica dentro de la literatura publicada. Se observa además que el rendimiento del clasificador entre las tres categorías alcanzó una medida macro-F1 (promedio de las medidas F1 de cada categoría) de 0,7248, un valor muy cercano a los 0.727 obtenidos por González y Martínez (2018), el antecedente con los mejores resultados sobre el corpus AnCora.

Para la creación de los clasificadores propuestos se extrajo y refinó la información contenida en el corpus AnCora para permitir la identificación de verbos con sujeto omitido e impersonales, anotándose manualmente la categoría de 1323 de verbos. Este conjunto de datos resulta por tanto más adecuado para la tarea de reconocer sujetos omitidos e impersonales y se espera que pueda ser utilizado en trabajos futuros, ya sea como corpus para el entrenamiento de una nueva herramienta o para comparación de resultados.

Los experimentos realizados validaron la factibilidad de adaptar un framework orientado a la traducción automática (OpenNMT) a una tarea de análisis lingüístico particular como es la detección de verbos con sujeto omitido en español. Mediante el uso de un corpus transformado para adaptar el proceso de traducción secuencia-a-secuencia a la generación de etiquetas con las categorías deseadas, se generaron clasificadores altamente efectivos en cuanto a performance (con tiempos de procesamiento $O(n^2)$ para entradas de largo n, frente a $O(n^3)$ en analizadores sintácticos tradicionales), con requisitos reducidos respecto al costo computacional del entrenamiento de los modelos (2,5 horas sobre una GPU *entry-level* de 2020) y obteniendo resultados de una calidad comparable a los presentados por estudios precedentes que constituyen el estado del arte para el reconocimiento del sujeto omitido en textos en idioma español.

Los resultados obtenidos demuestran que los frameworks de traducción automática no solo son eficaces para el análisis sintáctico generalista como lo demostró *Grammar as a Foreign Language* (Vinyals y cols., 2014) sino que también pueden emplearse de forma efectiva para problemas particulares (como la detección de sujeto tácito, el reconocimiento de verbos impersonales y el etiquetado de clases léxicas) de un modo flexible y eficiente aun ante recursos limitados en cuanto a hardware y tiempo de cómputo.

6.2. Trabajo futuro

La viabilidad identificada para la aplicación del enfoque de traducción automática para la tarea de reconocer sujetos omitidos sugiere que esta metodología podría aplicarse exitosamente a otras tareas de análisis lingüístico. Se identifican como potenciales problemas a abordar mediante este enfoque al reconocimiento de nombres propios, la detección de actitudes (opiniones positivas,

negativas, ironía) así como la generación de análisis sintácticos (*parsing*) que, si bien podrían presentar rendimientos de una calidad inferior a las líneas base del estado del arte, podrían resultar más eficientes computacionalmente.

En la sección de *Análisis de errores* se identifican varias estrategias para mitigar potencialmente la incidencia de algunas de las causas de clasificaciones erróneas encontradas en los clasificadores propuestos como la adición de un preprocesamiento que segmente cláusulas verbales y que incorpore información sobre el lema de cada palabra de forma que todas las formas inflecionales de un lexema (conjugaciones de un verbo, variantes en número y género de un sustantivo o adjetivo) se puedan identificar dentro del vocabulario y así disminuir el número de instancias para las que no se pudo obtener una clasificación (verbos "no detectados"). Si la implementación de este tipo de estrategias lograse corregir el número de verbos no detectados, se identifica un potencial para generar clasificadores capaces de identificar sujetos omitidos con una medida F1 superior a 0.8 y un *accuracy* general en el entorno de un 90 %, valores superiores a los actualmente encontrados en el estado del arte para el problema.

Se identifica asimismo el potencial para aplicar esta técnica en otras lenguas donde, como en el español, la detección de sujetos omitidos o usos impersonales resulta no trivial. En particular, el hecho de que el corpus AnCora elaborado por la Universitat de Barcelona posee también una versión en catalán permitiría replicar el estudio para este idioma.

Referencias

Azzinari, A., y Martínez, A. (2016, 06). *Representación de palabras en espacios de vectores.*

Brownlee, J. (2017). *What is the difference between test and validation datasets?* `https://machinelearningmastery.com/difference-test-validation-datasets/`. Machine Learning Mastery (blog).

Cardellino, C. (2019, August). *Spanish Billion Words Corpus and Embeddings.* Descargado de `https://crscardellino.github.io/SBWCE/`

Chiruzzo, L., y Wonsever, D. (2020, julio). Statistical deep parsing for Spanish using neural networks. En *Proceedings of the 16th international conference on parsing technologies and the iwpt 2020 shared task on parsing into enhanced universal dependencies* (pp. 132–144). Online: Association for Computational Linguistics. Descargado de `https://aclanthology.org/2020.iwpt-1.14` doi: 10.18653/v1/2020.iwpt-1.14

Christensen, A. (2023). *How many languages does chatgpt support?* `https://seo.ai/blog/how-many-languages-does-chatgpt-support`. SEO.ai (blog).

Cleary, J. G., y Trigg, L. E. (1995). K*: An instance-based learner using an entropic distance measure. En *12th international conference on machine learning* (p. 108-114).

Delor, M., Martí, A., y Recasens, M. (2008a, 01). Ancora: Multilevel annotated corpora for catalan and spanish. *Proceedings of 6th International Conference on Language Resources and Evaluation*, 96-101.

Delor, M., Martí, A., y Recasens, M. (2008b). *Corpus ancora.* `http://clic.ub.edu/corpus/es/ancora-descarregues`. Centre de Llenguatge i Computació, Universitat de Barcelona.

Delor, M., Martí, A., y Recasens, M. (2008c). *Note on morphology annotation.* `http://clic.ub.edu/corpus/webfm_send/18`. Centre de Llenguatge i Computació, Universitat de Barcelona.

Delua, J. (2021). *Supervised vs unsupervised learning: What's the difference?* `https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning`. IBM Cloud (blog).

Devlin, J., Chang, M., Lee, K., y Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, *abs/1810.04805*. Descargado de `http://arxiv.org/abs/1810.04805`

Dryer, M. S. (2013). Expression of pronominal subjects. En M. S. Dryer y M. Haspelmath (Eds.), *The world atlas of language structures online.* Leipzig: Max Planck Institute for Evolutionary Anthropology. Descargado de `https://wals.info/chapter/101`

Dugar, P. (2019). *Attention - seq2seq models.* `https://towardsdatascience.com/day-1-2-attention-seq2seq-models-65df3f49e263`. Towards Data Science (blog).

Ferrández, A., y Peral, J. (2000). A computational approach to zero-pronouns in spanish. En *Proceedings of the 38th annual meeting of the association for computational linguistics* (pp. 166–172).

Gandhi, R. (2018). *Naive bayes classifier.* `https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c`. Towards Data Science (blog).

Gerez, E., Irazusta, J. P., y Irace, M. (2019). *Detección y resolución de sujetos nulos en español.* Proyecto de grado. Uruguay. Universidad de la República.

González, L., y Martínez, V. (2018). *Detección de sujetos omitidos en el español.* Proyecto de grado. Uruguay. Universidad de la República.

Haspelmath, M. (2001, 01). The european linguistic area: Standard average european. *ResearchGate*, *2*.

Hochreiter, S., y Schmidhuber, J. (1997, 12). Long short-term memory. *Neural computation*, *9*, 1735-80. doi: 10.1162/neco.1997.9.8.1735

IBM. (2020a). *Supervised vs unsupervised learning: What's the difference?* `https://www.ibm.com/topics/machine-learning`. IBM Topics (blog).

IBM. (2020b). *What is overfitting?* `https://www.ibm.com/topics/overfitting`. IBM Topics (blog).

Klein, G., Kim, Y., Deng, Y., Senellart, J., y Rush, A. M. (2017). Opennmt: Open-source toolkit for neural machine translation. *CoRR*, *abs/1701.02810*. Descargado de `http://arxiv.org/abs/1701.02810`

Kornfield, L., y Kuguel, I. (2012). *El español rioplatense desde una perspectiva generativa.* Sociedad Argentina de Lingüística.

Masood, M., Nawaz, M., Malik, K. M., Javed, A., y Irtaza, A. (2021). Deepfakes generation and detection: State-of-the-art, open challenges, countermeasures, and way forward. *CoRR*, *abs/2103.00484*. Descargado de `https://arxiv.org/abs/2103.00484`

Matthews, P., y Matthews, P. (1997). *The concise oxford dictionary of linguistics.* Oxford University Press. Descargado de `https://books.google.com.uy/books?id=aYoYAAAAIAAJ`

Mikolov, T., Chen, K., Corrado, G., y Dean, J. (2013). *Efficient estimation of word representations in vector space.* arXiv. Descargado de `https://arxiv.org/abs/1301.3781` doi: 10.48550/ARXIV.1301.3781

Norvig, S. R. P. (2020). *Artificial intelligence: A modern approach* (4.ª ed.). Pearson.

Nyuytiymbiy, K. (2020). *Parameters and hyperparameters in machine learning and deep learning.* `https://towardsdatascience.com/parameters-and-hyperparameters-aa609601a9ac`. Towards Data Science (blog).

OpenNMT. (2020). *Documentación de opennmt.* https://opennmt.net/OpenNMT-py. opennmt.net.

Pennington, J., Socher, R., y Manning, C. (2014b, octubre). GloVe: Global vectors for word representation. En *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532–1543). Doha, Qatar: Association for Computational Linguistics. Descargado de https://aclanthology.org/D14-1162 doi: 10.3115/v1/D14-1162

Pennington, J., Socher, R., y Manning, C. D. (2014a). Glove: Global vectors for word representation. En *Emnlp* (Vol. 14, pp. 1532–1543).

Real Academia Española, R. A. E. (2011). *Nueva gramática básica de la lengua española.* Espasa.

Rello, L., Baeza-Yates, R., y Mitkov, R. (2012, abril). Elliphant: Improved automatic detection of zero subjects and impersonal constructions in Spanish. En *Proceedings of the 13th conference of the European chapter of the association for computational linguistics* (pp. 706–715). Avignon, France: Association for Computational Linguistics. Descargado de https://aclanthology.org/E12-1072

Rojas, J. P., Cañete, J., y Nguyen, D. (2018). *Spanish word embeddings.* https://github.com/dccuchile/spanish-word-embeddings. GitHub.

School of Information Berkeley. (2022). *What is machine learnign?* https://ischoolonline.berkeley.edu/blog/what-is-machine-learning/. School of Information, Berkeley (artículo).

Singhal, G. (2020). *Introduction to lstm units in rnn.* https://www.pluralsight.com/guides/introduction-to-lstm-units-in-rnn. Pluralsight (blog).

Stanovsky, G., y Dagan, I. (2018). Semantics as a foreign language. En *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 2412–2421).

Sutskever, I., Vinyals, O., y Le, Q. V. (2014). Sequence to sequence learning with neural networks. En Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, y K. Weinberger (Eds.), *Advances in neural information processing systems* (Vol. 27). Curran Associates, Inc. Descargado de https://proceedings.neurips.cc/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf

Suárez Palma, I. (2012). *La gramática generativa como fuente de aplicaciones en la enseñanza de lenguas extranjeras.* Tesis magistral. España. Universidad de Oviedo.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. *CoRR*, *abs/1706.03762*. Descargado de http://arxiv.org/abs/1706.03762

Vinyals, O., Kaiser, L., Koo, T., Petrov, S., Sutskever, I., y Hinton, G. (2014). Grammar as a foreign language. arXiv.

Yang, L. (2020). *Gpt-3, transformers and the wild world of nlp.* https://towardsdatascience.com/gpt-3-transformers-and-the-wild-world-of-nlp-9993d8bb1314. Towards Data Science (blog).

Anexo 1: Características del hardware empleado

Los modelos descritos en el presente reporte fueron entrenados y evaluados en un una laptop Acer Nitro 5 N18C3, equipo que presenta las siguientes características de hardware:

- **CPU:** Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz
- **RAM:** 8,00 GB
- **GPU:** Nvidia GeForce GTX 1650 @ 930-1395 MHz (2020)
- **Memoria dedicada de GPU:** 4,00 GB
- **Sistema operativo:** Windows 10 Home (x64)

El modelo de tarjeta gráfica (GTX 1650) se considera de *entry-level* dentro de la serie GeForce 16 del fabricante Nvidia.

Para modelos con las características planteadas en el trabajo, se observa experimentalmente que el hardware utilizado fue capaz de procesar un promedio de 20 000 pasos de entrenamiento en una hora para un modelo de dos capas ocultas LSTM con 500 unidades y en dos horas y media para un modelo con dos capas ocultas de 1000 unidades.

Anexo 2: Configuraciones de OpenNMT empleadas

Para el entrenamiento de un modelo de traducción automática utilizando el framework OpenNMT resulta necesario establecer, a través de un archivo YAML, una serie de configuraciones e hiperparámetros que definirán las características del modelo a entrenar, los corpus a utilizar para entrenar y evaluar los modelos y el hardware a utilizar, entre otros aspectos.

- **data**: Se define la ubicación en disco de los archivos de entrada y salida a emplear como corpus de entrenamiento y de validación, los cuales en este proyecto fueron generados con las particiones de entrenamiento y desarrollo (respectivamente) del corpus AnCora.

- **src vocab**, **tgt vocab**: Ubicación en disco del archivo de vocabulario generado a partir del corpus por el comando *onmt build vocab*.

- **world size**, **gpu ranks**: Define el número de procesos distribuidos en unidades de procesamiento gráfico (GPU) y su prioridad. En los experimentos de este estudio se empleó un *world size* con valor 1 dado que el entrenamiento se realizó sobre una sola GPU.

- **batch size**: Número de ejemplos procesados en paralelo durante el entrenamiento; un mayor *batch size* permite un entrenamiento más ágil pero requiere disponer de más memoria. Dado que la memoria dedicada de la GPU utilizada fue un factor limitante, para los modelos de mayor dimensionalidad se utilizó un valor de *batch size* relativamente bajo (8, frente a los 64 que OpenNMT maneja como valor predeterminado).

- **train steps**, **save checkpoint steps**: Cantidad de pasos (iteraciones sobre el corpus) a realizar durante el entrenamiento del modelo y número de pasos tras los cuales se guardará la configuración "intermedia" de tensores del modelo, permitiendo la evaluación del mismo en distintos puntos del entrenamiento. Con los modelos de menor dimensionalidad (hasta dos capas ocultas de 1000 unidades) se entrenó la red por 20 000 pasos, evaluando los resultados intermedios cada 2000. Para modelos mayores (en particular para los de tres capas ocultas) fue necesario prolongar el entrenamiento

hasta los 200 000 pasos antes de identificar que el modelo convergía, y se pasó a evaluar el rendimiento del modelo cada 10 000 pasos.

- **rnn size**: Cantidad de unidades por capa en las redes neuronales recurrentes (RNN; específicamente redes LSTM). Se reportan resultados para modelos con 500, 1000 y 1500 unidades por capa.
- **layers**: Número de capas ocultas del modelo. Se estudiaron modelos con 1, 2, 3 y 4 capas ocultas.
- **word vec size**: Dimensionalidad de la representación vectorial (*embedding*) empleada para los tokens en el modelo. En todos los experimentos se utilizó una dimensionalidad de 300.
- **embeddings type**, **src embeddings**: En los experimentos en los que se utilizó colecciones de *embeddings* preentrenados, estos parámetros definen el tipo (GloVe, Word2Vec) y ubicación del archivo de embeddings a utilizar. Ambos parámetros se omiten en los experimentos con *embeddings* aleatorios autogenerados por el framework.

Anexo 3: Identificación de verbos en el corpus AnCora

El corpus AnCora elaborado por la Universitat de Barcelona (Delor y cols., 2008a) (Delor y cols., 2008b) comprende un conjunto de cerca de 17 000 oraciones tomadas de 1635 artículos periodísticos anotados en base a su estructura sintáctica e información adicional como relaciones semánticas. Las oraciones anotadas se presentan como archivos XML que capturan la estructura arbórea de un análisis de constituyentes, donde cada nodo corresponde o bien a un elemento léxico (una palabra, una marca de puntuación, un número, fecha, etc) o a un grupo sintáctico. Cada nodo posee múltiples atributos que, dependiendo del caso, indicarán elementos como la clase gramatical, la palabra como string, el lema, características inflexionales (número, género, tiempo, aspecto, modo) y, potencialmente, las función sintáctica y semántica de un grupo nominal.

Para la identificación de sujetos omitidos mediante el enfoque de traducción empleado en este trabajo resultó necesario extraer las oraciones del corpus en dos formatos, correspondientes a la entrada y la salida esperada por los modelos propuestos:

- Como entrada, se utilizan los elementos léxicos que componen la oración. Para asegurar que OpenNMT reconozca los tokens correctamente y evitar que las diferencias en el uso de mayúscula amplíen innecesariamente el número vocabulario a considerar, los elementos léxicos (palabras, puntuación) se presentan en los archivos de entrada en minúscula y separados por espacios (incluso previo a marcas de puntuación como puntos y comas). A modo de ejemplo "Creo que Juan estaba leyendo un libro hace unos días." se presentará como *"creo que juan estaba leyendo un libro hace unos días .".*
- Como salida, se optó por utilizar las etiquetas de clase gramatical (*part of speech*, POS) registradas en AnCora para cada elemento léxico pero con etiquetas especiales para indicar diferentes escenarios respecto a la presencia o ausencia de sujeto en los verbos:
    - Se emplea una etiqueta especial (**i**) para los verbos no finitos (infinitivos, gerundios y participios) para los cuales no aplica el concepto de sujeto en español.

- Los verbos finitos con sujeto omitido se marcan como distintos a los que presentan sujeto explícito, ya sea mediante la inclusión de una marca **0** en la ubicación tentativa donde pudo ubicarse el sujeto ("enfoque de marca de sujeto omitio") o mediante el uso de etiquetas distintas para verbos con o sin sujeto explícito (**v** y **w** respectivamente, "enfoque de clasificación de verbos").
- En los experimentos donde se distingue el caso de verbos impersonales, los mismos se anotan con otra etiqueta, **y**.

A modo de ejemplo "Creo que Juan estaba leyendo un libro hace unos días." (donde "creo" presenta sujeto omitido, "estaba" sujeto explícito, "leyendo" es un gerundio y "hace" aparece en un uso impersonal) se presentará como *"**w** c n **v** **i** d n **y** d n f"*.

Si bien la generación de los archivos de entrada es trivial (basta con recorrer los archivos XML y extraer el contenido de los nodos correspondientes a elementos léxicos, donde la palabra a mostrar se lista como el atributo *wd*, por "word"), la generación del archivo de salida presentó la dificultad de reconocer los diferentes escenarios para los verbos, para lo cual se emplearon las siguientes técnicas:

- Los elementos léxicos correspondientes a verbos pueden identificarse en los archivos de AnCora por utilizar nodos con la etiqueta **v**. Los mismos incluyen tanto verbos finitos como no finitos.
- Los nodos correspondientes a verbos usualmente presentan un atributo *pos* con un string de siete caracteres (etiqueta EAGLES) que representa las categorías gramaticales codificadas por el verbo, como su modo, tiempo verbal y concordancia en persona y número con su sujeto (independientemente de si este es explícito, omitido o si el verbo es impersonal). Se identificó que los verbos no finitos (infinitivos, participios y gerundios) podían caracterizarse por un valor nulo de una marca de persona indicada en el quinto carácter de dicha etiqueta por lo que se procedió a etiquetar como formas no finitas **i** a todo verbo cuya etiqueta *pos* presente un 0 en su quinta posición.
- Para la mayor parte de los verbos finitos con sujeto explícito, el sintagma nominal (nodo no-final con etiqueta **sn**) correspondiente a su sujeto presenta el atributo *func* (función sintáctica) con el valor *suj*. En caso de que se detecte dicho campo en un nodo relacionado a un verbo finito, el mismo puede etiquetarse como "con sujeto explícito", **v**.
- La mayoría de las oraciones con verbos personales con sujeto omitido presentan un nodo de sintagma nominal vacío (nodo final con etiqueta **sn**) con el atributo *elliptic* (elipsis de sujeto) el cual representa una ubicación hipotética (usualmente precediendo al verbo) en la que pudo ubicarse el sujeto de habérselo explicitado. En el "enfoque de marca de sujeto omitido", este nodo (que no aparecerá en la entrada por no tener un elemento

léxico asociado) se representa en la salida con la marca **0** mientras que en el "enfoque de clasificación de verbos" la detección de este nodo en la vecindad de un nodo de verbo finito lleva a que se etiqueta al segundo como "verbo con sujeto omitido", **w**.

- Los verbos impersonales en oraciones reflejas (precedidos del pronombre "se") suelen contener un nodo de "morfema verbal" con el atributo *func* con valor "*impers*" que permite reconocer al verbo finito contenido dentro de dicho nodo como impersonal, **y**.

Estas técnicas permiten extraer del corpus la clasificación correcta para un 94 % de sus verbos; el 6 % restante (1323 verbos finitos) no poseen marcas que identifiquen su clase inequívocamente. Un estudio sobre estos verbos demostró que estos casos correspondían a los siguientes escenarios:

- Verbos impersonales en construcciones donde no se encuentran precedidos por el pronombre "se"; especialmente instancias del verbo "haber" ("hay", "hubo"), del verbo "hacer" ("hace muchos años", "hace frío") y del verbo "ser" ("es que..."). Las instancias impersonales del verbo haber representaron cerca de la mitad de los verbos sin clasificación.

- Pares de verbos (de cualquier categoría) separados por conjunciones dentro de una frase relativa, como "(la persona) que vive y lucha". Estos casos representan inconsistencias en la anotación del corpus, donde usualmente en las conjunciones como "María canta y baila" el primer verbo contiene marcas de sujeto (explícito u omitido, según el caso) mientras que los demás verbos se anotan como con sujeto omitido ("María canta y (ella) baila"). Aplicando el criterio general de AnCora, el ejemplo "que vive y lucha" debería haber indicado un sujeto explícito para "vive" (el pronombre "que") y sujeto omitido para "lucha".

- Una cantidad muy limitada de verbos con sujeto explícito no etiquetado como tal separado del verbo correspondiente por una acotación, comentario entre paréntesis o similar.

Para asignar marcas a parte de estos verbos se procedió a tomar el criterio que las formas del verbo "haber" sin marcas se consideran como impersonales a menos que les siguiere un participio. Las instancias restantes fueron etiquetadas manualmente siguiendo criterios análogos a los utilizados en otras instancias de AnCora para asegurar la consistencia en el formato del corpus.

Anexo 4: Resultados experimentales

A continuación se presentan en mayor detalle los resultados obtenidos durante el desarrollo del proyecto por los distintos modelos entrenados para este estudio.

Todos los resultados fueron evaluados sobre la partición de desarrollo del corpus AnCora salvo cuando se indique lo contrario.

.1. Modelos de clasificación binaria

Los modelos en esta sección se entrenaron para resolver el problema de detectar verbos sin sujeto explícito ("con sujeto omitido", aunque sin distinguir el caso de verbos impersonales) frente a verbos con sujeto explícito, por lo que únicamente se calculan métricas para la primera categoría.

Dado que las salidas en los modelos utilizados puede no contener etiquetas de clasificación para todos los verbos en su entrada, se distinguen dos tipos de resultados erróneos: las clasificaciones en la categoría incorrecta y las no detecciones. Ambos tipos se consideran como "falsos negativos" o "falsos positivos", según el caso, a efectos del cálculo de métricas de *precision*, *recall* y medida F1.

.1.1. Enfoque de marcas de sujeto omitido

Experimentos utilizando una etiqueta pseudo-POS para marcar la posición hipotética del un sujeto omitido; sección 5.2.1. Matrices de confusión y métricas en tabla 1.

Modelo 1

- **Formato de salida**: etiquetas POS, con marcas **0** para indicar la posición hipotética de un sujeto omitido (según criterio de AnCora).
- **Dimensiones del modelo**: 1 capa de 500 unidades.
- **Representación de tokens**: *embeddings* autogenerados por OpenNMT

- **Entrenamiento del modelo**: 6 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{0}$,5 horas para 20 000 iteraciones.

Modelo 2

- **Formato de salida**: etiquetas POS, con marcas **0** para indicar la posición hipotética de un sujeto omitido (según criterio de AnCora).
- **Dimensiones del modelo**: 2 capas de 500 unidades.
- **Representación de tokens**: *embeddings* autogenerados por OpenNMT
- **Entrenamiento**: 10 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{1}$ hora para 20 000 iteraciones.

Modelo 3

- **Formato de salida**: etiquetas POS, con marcas **0** para indicar la posición hipotética de un sujeto omitido (según criterio de AnCora).
- **Dimensiones del modelo**: 1 capa de 500 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento**: 4 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{0}$,5 horas para 20 000 iteraciones.

Modelo 4

- **Formato de salida**: etiquetas POS, con marcas **0** para indicar la posición hipotética de un sujeto omitido (según criterio de AnCora).
- **Dimensiones del modelo**: 2 capas de 500 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento**: 10 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{1}$ hora para 20 000 iteraciones.

.1.2. Enfoque de clasificación de verbos - comparación de embeddings

Experimentos utilizando etiquetas distintas para verbos con y sin sujeto explícito, comparando el rendimiento del modelo con diferentes juegos de embeddings; secciones 5.2.1 y. 5.2.2 Matrices de confusión y métricas en tablas 2 y 3.

Modelo 1	1 capa × 500 unidades	Embeddings autogenerados	6 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**539 (51,53 %)**	389 (37,19 %)	389 (37,19 %)
Sujeto explícito	189 (6,36 %)	**2571 (86,48 %)**	2571 (86,48 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
76,73 %	74,04 %	51,53 %	**60,77 %**
Modelo 2	2 capas × 500 unidades	Embeddings autogenerados	10 000 pasos
Categoría real	Detectado como sujeto omitido	Detectado como sujeto explícito	No detectado
Sujeto omitido	**678 (64,82 %)**	292 (27,92 %)	292 (27,92 %)
Sujeto explícito	262 (8,81 %)	**2581 (86,81 %)**	2581 (86,81 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
79,39 %	72,13 %	64,82 %	**68,28 %**
Modelo 3	1 capa × 500 unidades	Embeddings GloVe SBWC	4 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**492 (47,04 %)**	394 (37,67 %)	394 (37,67 %)
Sujeto explícito	231 (7,77 %)	**2468 (83,01 %)**	2468 (83,01 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
67,60 %	68,05 %	47,04 %	**55,62 %**
Modelo 4	2 capas × 500 unidades	Embeddings GloVe SBWC	10 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**712 (68,07 %)**	250 (23,90 %)	250 (23,90 %)
Sujeto explícito	239 (8,04 %)	**2597 (87,35 %)**	2597 (87,35 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
81,66 %	74,87 %	68,07 %	**71,31 %**

Tabla 1: Resultados de modelos 1 a 4 (modelos con el enfoque de marcas **0** para sujetos omitidos).

Modelo 5

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 1 capa de 500 unidades.
- **Representación de tokens**: *embeddings* autogenerados por OpenNMT
- **Entrenamiento del modelo**: 14 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{0}$,5 horas para 20 000 iteraciones.

Modelo 6

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 2 capas de 500 unidades.
- **Representación de tokens**: *embeddings* autogenerados por OpenNMT
- **Entrenamiento del modelo**: 16 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{1}$ hora para 20 000 iteraciones.

Modelo 7

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 2 capas de 1000 unidades.
- **Representación de tokens**: *embeddings* autogenerados por OpenNMT
- **Entrenamiento del modelo**: 6 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{2}$,5 horas para 10 000 iteraciones.

Modelo 8

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 1 capa de 500 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 6 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{0}$,5 horas para 20 000 iteraciones.

Modelo 9

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 2 capas de 500 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 6 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{1}$ hora para 20 000 iteraciones.

Modelo 10

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 2 capas de 1000 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 12 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{2}$,5 horas para 20 000 iteraciones.

Modelo 11

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 2 capas de 500 unidades.
- **Representación de tokens**: *embeddings* Word2Vec del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 18 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{1}$ hora para 20 000 iteraciones.

Modelo 12

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 2 capas de 1000 unidades.
- **Representación de tokens**: *embeddings* Word2Vec del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 14 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{2}$,5 horas para 20 000 iteraciones.

Modelo 5	1 capa × 500 unidades	Embeddings autogenerados	14 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**330 (31,55 %)**	498 (47,61 %)	498 (47,61 %)
Sujeto explícito	141 (4,74 %)	**2392 (80,46 %)**	2392 (80,46 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
60,18 %	70,06 %	31,55 %	**43,51 %**
Modelo 6	2 capas × 500 unidades	Embeddings autogenerados	16 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**770 (73,61 %)**	208 (19,89 %)	208 (19,89 %)
Sujeto explícito	345 (11,60 %)	**2487 (83,65 %)**	2487 (83,65 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
79,59 %	69,06 %	73,61 %	**71,26 %**
Modelo 7	2 capas × 1000 unidades	Embeddings autogenerados	10 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**713 (68,16 %)**	270 (25,81 %)	270 (25,81 %)
Sujeto explícito	247 (8,31 %)	**2607 (87,69 %)**	2607 (87,69 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
80,86 %	74,27 %	68,16 %	**71,09 %**
Modelo 8	1 capa × 500 unidades	Embeddings GloVe SBWC	6 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**95 (9,08 %)**	365 (34,89 %)	365 (34,89 %)
Sujeto explícito	99 (3,33 %)	**1714 (57,65 %)**	1714 (57,65 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
44,53 %	48,97 %	9,08 %	**15,32 %**
Modelo 9	2 capas × 500 unidades	Embeddings GloVe SBWC	6 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**781 (74,67 %)**	171 (16,35 %)	171 (16,35 %)
Sujeto explícito	288 (9,69 %)	**2490 (83,75 %)**	2490 (83,75 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
80,79 %	73,06 %	74,67 %	**73,85 %**
Modelo 10	2 capas × 1000 unidades	Embeddings GloVe SBWC	12 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**753 (71,99 %)**	205 (19,60 %)	205 (19,60 %)
Sujeto explícito	214 (7,20 %)	**2593 (87,22 %)**	2593 (87,22 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
82,88 %	77,87 %	71,99 %	**74,81 %**

Tabla 2: Resultados de modelos 5 a 10 (modelos con enfoque de marcas **v** y **w** para verbos con y sin sujetos explícito).

Modelo 13

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 2 capas de 500 unidades.
- **Representación de tokens**: *embeddings* Word2Vec de la colección emb39.
- **Entrenamiento del modelo**: 10 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{1}$ hora para 20 000 iteraciones.

Modelo 14

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 2 capas de 1000 unidades.
- **Representación de tokens**: *embeddings* Word2Vec de la colección emb39.
- **Entrenamiento del modelo**: 20 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: 12 horas para 100 000 iteraciones (el modelo no convergió hasta las 70 000 iteraciones).

.1.3. Enfoque de clasificación de verbos - modelos de mayor dimensionalidad

Experimentos utilizando etiquetas distintas para verbos con y sin sujeto explícito con más de dos capas o más de 1000 unidades por capa; sección 5.2.3 Matrices de confusión y métricas en tabla 4.

Modelo 15

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 2 capas de 1500 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 100 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{1}$4 horas para 100 000 iteraciones.

Modelo 11	2 capas × 500 unidades	Embeddings Word2Vec SBWC	18 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**688 (65,77 %)**	253 (24,19 %)	253 (24,19 %)
Sujeto explícito	227 (7,64 %)	**2544 (85,57 %)**	2544 (85,57 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
80,04 %	75,19 %	65,77 %	**70,17 %**

Modelo 12	2 capas × 1000 unidades	Embeddings Word2Vec SBWC	14 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**678 (64,82 %)**	278 (26,58 %)	278 (26,58 %)
Sujeto explícito	207 (6,96 %)	**2607 (87,69 %)**	2607 (87,69 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
81,13 %	76,61 %	64,82 %	**70,22 %**

Modelo 13	2 capas × 500 unidades	Embeddings Word2Vec emb39	10 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**697 (66,63 %)**	252 (24,09 %)	252 (24,09 %)
Sujeto explícito	253 (8,51 %)	**2535 (85,27 %)**	2535 (85,27 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
80,06 %	73,37 %	66,63 %	**69,84 %**

Modelo 14	2 capas × 1000 unidades	Embeddings Word2Vec emb39	20 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**702 (67,11 %)**	252 (24,09 %)	252 (24,09 %)
Sujeto explícito	201 (6,76 %)	**2597 (87,35 %)**	2597 (87,35 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
81,52 %	77,74 %	67,11 %	**72,04 %**

Tabla 3: Resultados de modelos 11 a 14 (modelos con enfoque de marcas **v** y **w** para verbos con y sin sujetos explícito).

Modelo 16

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 3 capas de 500 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 18 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{1}$ hora para 20 000 iteraciones.

Modelo 17

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 3 capas de 1000 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 90 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{1}$0,5 horas para 100 000 iteraciones.

Modelo 18

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 3 capas de 1500 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 100 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{1}$9,5 horas para 100 000 iteraciones.

Modelo 19

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 4 capas de 500 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 80 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{4}$,5 horas para 100 000 iteraciones.

Modelo 20

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito o **w** para sujeto omitido.
- **Dimensiones del modelo**: 4 capas de 1000 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 90 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: ~10,5 horas para 100 000 iteraciones.

.2. Modelos de clasificación ternaria

Los modelos en esta sección se entrenaron para resolver el problema de distinguir entre tres escenarios para cada verbo finito: que presente sujeto explícito, que presente sujeto omitido o que presente un uso impersonal. Se presentan métricas para cada clasificación.

Dado que las salidas en los modelos utilizados puede no contener etiquetas de clasificación para todos los verbos en su entrada, se distinguen dos tipos de resultados erróneos: las clasificaciones en la categoría incorrecta y las no detecciones. Ambos tipos se consideran como "falsos negativos" o "falsos positivos", según el caso, a efectos del cálculo de métricas de *precision*, *recall* y medida F1.

Experimentos utilizando diferentes etiquetas para las tres categorías consideradas para los verbos finitos (**v** para sujeto explícito,**w** para sujeto omitido e **y** para uso impersonal). Ver sección; sección 5.3. Matrices de confusión y métricas en tabla 5.

Modelo 21

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito, **w** para sujeto omitido o **y** para verbos impersonales.
- **Dimensiones del modelo**: 2 capas de 500 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 10 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: ~1 hora para 20 000 iteraciones.

Modelo 15	2 capas × 1500 unidades	Embeddings GloVe SBWC	100 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**712 (68,07 %)**	246 (23,52 %)	246 (23,52 %)
Sujeto explícito	155 (5,21 %)	**2646 (89,00 %)**	2646 (89,00 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
83,22 %	82,12 %	68,07 %	**74,44 %**
Modelo 16	3 capas × 500 unidades	Embeddings GloVe SBWC	18 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**759 (72,56 %)**	195 (18,64 %)	195 (18,64 %)
Sujeto explícito	233 (7,84 %)	**2563 (86,21 %)**	2563 (86,21 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
82,31 %	76,51 %	72,56 %	**74,48 %**
Modelo 17	3 capas × 1000 unidades	Embeddings GloVe SBWC	90 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**769 (73,52 %)**	195 (18,64 %)	195 (18,64 %)
Sujeto explícito	213 (7,16 %)	**2613 (87,89 %)**	2613 (87,89 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
83,42 %	78,31 %	73,52 %	**75,84 %**
Modelo 18	3 capa × 1500 unidades	Embeddings GloVe SBWC	100 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**704 (67,30 %)**	258 (24,67 %)	258 (24,67 %)
Sujeto explícito	161 (5,42 %)	**2654 (89,27 %)**	2654 (89,27 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
83,20 %	81,39 %	67,30 %	**73,68 %**
Modelo 19	4 capas × 500 unidades	Embeddings GloVe SBWC	80 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**457 (43,69 %)**	383 (36,62 %)	383 (36,62 %)
Sujeto explícito	353 (11,87 %)	**2232 (75,08 %)**	2232 (75,08 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
63,18 %	56,42 %	43,69 %	**49,25 %**
Modelo 20	4 capas × 1000 unidades	Embeddings GloVe SBWC	90 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**486 (46,46 %)**	411 (39,29 %)	411 (39,29 %)
Sujeto explícito	316 (10,63 %)	**2428 (81,67 %)**	2428 (81,67 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
67,72 %	60,60 %	46,46 %	**52,60 %**

Tabla 4: Resultados de modelos 15 a 20 (modelos con enfoque de marcas **v** y **w** para verbos con y sin sujetos explícito).

Modelo 22

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito, **w** para sujeto omitido o **y** para verbos impersonales.
- **Dimensiones del modelo**: 2 capas de 1000 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 12 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: $\tilde{2}$,5 horas para 20 000 iteraciones.

Modelo 23

- **Formato de salida**: etiquetas POS, con marcas **v** para verbos con sujeto explícito, **w** para sujeto omitido o **y** para verbos impersonales.
- **Dimensiones del modelo**: 3 capas de 1000 unidades.
- **Representación de tokens**: *embeddings* GloVe del Spanish Billion Word Corpus (SBWC).
- **Entrenamiento del modelo**: 110 000 iteraciones sobre el corpus.
- **Tiempo de entrenamiento**: 16,5 horas para 100 000 iteraciones.

.3. Evaluación final

En las tablas 6 y 7 se presentan las matrices de confusión y métricas de los modelos que presentaron el mayor rendimiento (modelos 9, 10 y 17 para clasificación binaria, modelos 21 y 22 para clasificación ternaria) evaluando sobre el corpus de testing.

Modelo 21	2 × 500 unidades	Emb. GloVe SBWC	10 000 pasos	Clasif. ternaria	Dev. corpus
Categoría real	Detecta expl.	Detecta omit.	Detecta imp.	No detectado	Total
Sujeto explícito	**2503 (86,13 %)**	170 (6,16 %)	19 (0,65 %)	205 (7,05 %)	2906
Sujeto omitido	158 (14,88 %)	**797 (75,05 %)**	6 (0,56 %)	101 (9,51 %)	1062
Verbo impersonal	16 (32,00 %)	14 (28,00 %)	**20 (40,00 %)**	0 (0 %)	50
Precision	90,50 %	80,51 %	44,44 %	**Macro accuracy**	**82,63 %**
Recall	86,13 %	75,05 %	40,00 %	**Macro F1**	**69,82 %**
Medida F1	89,67 %	77,68 %	42,11 %	**F1 ponderado**	**85,91 %**
Modelo 22	2 × 1000 unidades	Emb. GloVe SBWC	12 000 pasos	Clasif. ternaria	Dev. corpus
Categoría real	Detecta expl.	Detecta omit.	Detecta imp.	No detectado	Total
Sujeto explícito	**2591 (89,16 %)**	157 (5,40 %)	19 (0,65 %)	139 (4,78 %)	2906
Sujeto omitido	182 (17,14 %)	**792 (74,58 %)**	7 (0,66 %)	81 (7,63 %)	1062
Verbo impersonal	22 (44,00 %)	7 (14,00 %)	**21 (42,00 %)**	0 (0 %)	50
Precision	92,70 %	82,85 %	44,68 %	**Macro accuracy**	**84,72 %**
Recall	89,16 %	74,58 %	42,00 %	**Macro F1**	**70,90 %**
Medida F1	90,90 %	78,49 %	43,30 %	**F1 ponderado**	**87,03 %**
Modelo 23	3 × 1000 unidades	Emb. GloVe SBWC	110 000 pasos	Clasif. ternaria	Dev. corpus
Categoría real	Detecta expl.	Detecta omit.	Detecta imp.	No detectado	Total
Sujeto explícito	**2496 (85,89 %)**	260 (8,95 %)	12 (0,41 %)	138 (4,75 %)	2906
Sujeto omitido	308 (29,00 %)	**649**	3 (0,28 %)	102 (9,60 %)	1062
Verbo impersonal	32 (64,00 %)	11 (22,00 %)	**6 (12,00 %)**	1 (2,00 %)	50
Precision	88,01 %	70,54 %	28,57 %	**Macro accuracy**	**78,42 %**
Recall	85,89 %	61,11 %	12,00 %	**Macro F1**	**56,44 %**
Medida F1	86,94 %	65,49 %	16,90 %	**F1 ponderado**	**80,40 %**

Tabla 5: Resultados de modelos 21 a 23 (clasificación ternaria, incorporando reconocimiento de verbos impersonales).

Modelo 9	2 capas × 500 unidades	Embeddings GloVe SBWC	6 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**744 (70,86 %)**	215 (20,48 %)	215 (20,48 %)
Sujeto explícito	288 (9,25 %)	**2624 (84,29 %)**	2624 (84,29 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
80,52 %	72,09 %	70,86 %	**71,47 %**
Modelo 10	2 capas × 1000 unidades	Embeddings GloVe SBWC	12 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**731 (69,62 %)**	234 (22,29 %)	234 (22,29 %)
Sujeto explícito	221 (7,10 %)	**2709 (87,02 %)**	2709 (87,02 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
82,20 %	76,79 %	69,62 %	**73,03 %**
Modelo 17	3 capas × 1000 unidades	Embeddings GloVe SBWC	90 000 pasos
Categoría real	Detecta sujeto omitido	Detecta sujeto explícito	No detectado
Sujeto omitido	**761 (72,48 %)**	224 (21,33 %)	224 (21,33 %)
Sujeto explícito	214 (6,87 %)	**2752 (88,40 %)**	2752 (88,40 %)
Accuracy general	*Precision* (omitidos)	*Recall* (omitidos)	**F1** (omitidos)
83,80 %	78,05 %	72,48 %	**75,16 %**

Tabla 6: Resultados de modelos de mejor rendimiento para clasificación binaria evaluando sobre el corpus de testing.

Modelo 21	2 × 500 unidades	Emb. GloVe SBWC	10 000 pasos	Clasif. ternaria	Test corpus
Categoría real	Detecta expl.	Detecta omit.	Detecta imp.	No detectado	Total
Sujeto explícito	**2636 (86,63 %)**	187 (6,15 %)	15 (0,49 %)	205 (6,74 %)	3043
Sujeto omitido	158 (14,88 %)	**790 (74,39 %)**	8 (0,75 %)	106 (9,98 %)	1062
Verbo impersonal	13 (22,81 %)	17 (29,82 %)	**27 (47,37 %)**	0 (0 %)	57
Precision	93,91 %	79,48 %	54,00 %	**Macro accuracy**	**82,96 %**
Recall	86,63 %	74,39 %	47,37 %	**Macro F1**	**72,48 %**
Medida F1	90,12 %	76,85 %	50,47 %	**F1 ponderado**	**86,19 %**

Modelo 22	2 × 1000 unidades	Emb. GloVe SBWC	12 000 pasos	Clasif. ternaria	Test corpus
Categoría real	Detecta expl.	Detecta omit.	Detecta imp.	No detectado	Total
Sujeto explícito	**2707 (88,96 %)**	173 (5,69 %)	18 (0,59 %)	145 (4,77 %)	3043
Sujeto omitido	201 (18,93 %)	**769 (72,41 %)**	11 (1,04 %)	81 (7,63 %)	1062
Verbo impersonal	19 (33,33 %)	12 (21,05 %)	**26 (45,61 %)**	0 (0 %)	57
Precision	92,48 %	80,61 %	47,27 %	**Macro accuracy**	**84,14 %**
Recall	88,96 %	72,41 %	45,61 %	**Macro F1**	**71,14 %**
Medida F1	90,69 %	76,29 %	46,43 %	**F1 ponderado**	**86,41 %**

Tabla 7: Resultados de modelos de mejor rendimiento para clasificación ternaria evaluando sobre el corpus de testing.