Escuela Politécnica Superior de Jaén
Máster Universitario en Ingeniería Informática

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior (Jaén)

Trabajo Fin de Máster

INTEGRACIÓN DE CARACTERÍSTICAS LINGÜÍSTICAS EN REDES NEURONALES PROFUNDAS

Alumno/a: **Mármol Romero, Alba María**

Tutor/a: Prof. D. Arturo Montejo Ráez
Dpto.: Informática

Diciembre, 2022

Informe del Tutor/a del Trabajo Fin de Grado/Máster

Autor (Apellido1--Apellido2, Nombre)			
MÁRMOL ROMERO, ALBA MARÍA			
Título del Trabajo			
INTEGRACIÓN DE CARACTERÍSTICAS LINGÜÍSTICAS EN REDES NEURONALES PROFUNDAS.			
Titulación	MÁSTER EN INGENIERÍA INFORMÁTICA	Especialidad/ Mención	
Centro	ESCUELA POLITÉCNICA SUPERIOR DE JAÉN	Departamento	INFORMÁTICA
Tutor/a del TFG/TFM		Universidad/Institución	
ARTURO MONTEJO RÁEZ		UNIVERSIDAD DE JAÉN	

Resumen Castellano (máx. 150 palabras)

El Trabajo de Fin de Máster consiste en la integración de características lingüísticas en redes neuronales profundas. Para ello se ha implementado un conjunto de sistemas capaces de detectar si un sujeto padece rasgos de ludopatía o no, teniendo en cuenta su historial de publicaciones en redes sociales. Estos sistemas desarrollados se basan en el uso de embeddings de una red Transformer junto con características lingüísticas relacionadas con la volumetría, diversidad léxica, complejidad léxica y puntuaciones relacionadas con las emociones. Para la realización de este trabajo se ha requerido de la implementación de código en lenguaje Python y el uso de herramientas para sistemas de aprendizaje automático. Los datos utilizados y la forma de evaluación se apoyan en una campaña de evaluación. En la evaluación de los sistemas se comparan los resultados obtenidos de un sistema que no tiene en cuenta características lingüísticas con otros sistemas que sí las utilizan.

Resumen Inglés (máx. 150 palabras)

This end of Master's dissertation focuses on the integration of linguistic features in deep neural networks. To do this, a number of systems have been implemented that predict if a subject suffers from pathological gambling behaviour or not, taking into account his or her history of publications on social networks. These developed systems are based on the use of sentence embeddings from Transformers with features related to volumetry, lexical diversity, complexity metrics, and emotion related scores. This work required the implementation of code in Python and the use of tools for machine learning systems. The data used and the form of evaluation are based on an evaluation campaign. The evaluation of the systems compares the results obtained with a system that does not take linguistic features with others systems that do.

Nomenclatura Internacional de Unesco para la Ciencia y Tecnología(http://skos.um.es/unesco6/)

Códigos UNESCO	Descriptor castellano	Descriptor Inglés
1203.04	INTELIGENCIA ARTIFICIAL	ARTIFICIAL INTELLIGENCE
1203.17	INFORMÁTICA	INFORMATICS

Agradecimientos

En primer lugar, me gustaría agradecer a mi familia, especialmente a mi madre y a mi padre, por todo el amor y apoyo que me han dado desde siempre, de forma incondicional, ya que sin ellos llegar hasta aquí no hubiese sido posible. También agradecer a mis dos hermanos que hayan estado conmigo en este camino.

Por otro lado, quería agradecer a mis amigos y amigas, que se han alegrado de mis logros como si fuesen suyos. Y a mis compañeros de trabajo, que han hecho que cada día fuese un buen día. A todos ellos, por ayudarme a ver el lado divertido de las cosas incluso en los momentos más estresantes.

Además, agradecer a mis profesores del Máster por compartir su conocimiento y experiencia y acompañarme en esta etapa.

Finalmente, dar las gracias a mi tutor, Arturo Montejo, por guiarme durante este Trabajo de Fin de Máster, aconsejarme y resolver todas las dudas que se me presentaran durante estos meses. Agradecerle su paciencia, sus ánimos para seguir adelante y su motivación por continuar aprendiendo cada día.

Índice

Índice de figuras

Índice de tablas

Capítulo 1

Resumen.

El Trabajo de Fin de Máster del que trata este documento consiste en la integración de características lingüísticas en redes neuronales profundas. Para ello se ha implementado un conjunto de sistemas capaces de detectar si un sujeto padece rasgos de ludopatía o no, teniendo en cuenta su historial de publicaciones en redes sociales. Estos sistemas desarrollados se basan en el uso de *embeddings* de una red *Transformer* junto con características lingüísticas relacionadas con la volumetría, diversidad léxica, complejidad léxica y puntuaciones relacionadas con las emociones. Para la realización de este trabajo se ha requerido de la implementación de código en lenguaje Python y el uso de herramientas para sistemas de aprendizaje automático. Los datos utilizados y la forma de evaluación se apoyan en una campaña de evaluación. En la evaluación de los sistemas se comparan los resultados obtenidos de un sistema que no tiene en cuenta características lingüísticas con otros sistemas que sí las utilizan.

ABSTRACT

This end of Master's dissertation focuses on the integration of linguistic features in deep neural networks. To do this, a number of systems have been implemented that predict if a subject suffers from pathological gambling behaviour or not, taking into account his or her history of publications on social networks. These developed systems are based on the use of sentence embeddings from Transformers with features related to volumetry, lexical diversity, complexity metrics, and emotion related scores. This work required the implementation of code in Python and the use of tools for machine learning systems. The data used and the form of evaluation are based on an evaluation campaign. The evaluation of the systems compares the results obtained with a system that does not take linguistic features with others systems that do.

Capítulo 2

Introducción.

Este documento es la memoria de un Trabajo de Fin de Máster titulado "Integración de características lingüísticas en redes neuronales profundas" realizado para el Máster Universitario de Ingeniería Informática[1] de la Escuela Politécnica Superior de Jaén[2].

En este capítulo se desarrollará una pequeña introducción acerca del contenido del proyecto. Para ello, se comienza explicando la motivación por la que surgió esta idea y que impulsa al desarrollo de un sistema de redes neuronales profundas con integración de características lingüísticas. Seguidamente se mencionarán los objetivos principales y generales que se pretenden y desean alcanzar durante el desarrollo del proyecto y, en último lugar, se hará una pequeña introducción acerca de la estructura de la memoria presente y explicación breve de cada uno de los capítulos contenidos en de este documento.

2.1. Motivación.

La idea de este trabajo surgió motivada, principalmente, por la pasión y el interés que despierta en mí el Procesamiento del Lenguaje Natural (PLN) y su aplicación al ámbito social y de la salud. En segundo lugar, la motivación se apoya en la falta de experimentación respecto a la integración de distintas características lingüísticas, estudiadas en el campo del PLN, en redes neuronales. Actualmente está en auge el uso y aplicación de distintos tipos de redes neuronales profundas, sin embargo, la experimentación con la integración de distintos elementos en éstas es reducida por lo que hay un desconocimiento acerca de los resultados que dicha integración puede producir.

Los últimos avances relativos a la investigación con referencia al PLN han estado dominados por el uso de modelos de lenguaje del tipo *Transformer*. Por ello, en este documento pretendo hacer uso de esta arquitectura, que ha demostrado conseguir unos resultados superiores con

[1] https://eps.ujaen.es/masteres-oficiales/master-universitario-en-ingenieria-informatica
[2] https://eps.ujaen.es/

respecto a otro tipo de técnicas y arquitecturas, y generar un sistema apoyado en él, que facilite los objetivos que estamos buscando.

Se pretende determinar si dicha integración de características lingüísticas, tales como volumetría, diversidad o complejidad léxica, mejoran los sistemas. Para ello, se toma de partida una competición científica, consistente en el desarrollo de un sistema capaz de detectar precozmente signos patológicos de ludopatía en Internet, concretamente en redes sociales. Todo ello será detallado en profundidad en los capítulos siguientes.

2.2. Objetivos.

A continuación se enumeran los distintos objetivos, generales y específicos, presentes en este trabajo, siendo el objetivo principal la experimentación respecto al ámbito de la integración de características lingüísticas en redes neuronales profundas.

Objetivos específicos.

En esta parte de la sección, se enumeran los distintos objetivos generales presentes en este trabajo de fin de máster.

- Conocer el proceso de experimentación científica.
- Conocer y desarrollar soluciones de Procesamiento del Lenguaje Natural mediante redes *Transformer*.
- Desarrollar redes propias con integración de características externas.
- Evaluar y comparar las estrategias exploradas con el estado de la cuestión.
- Confeccionar la memoria del Trabajo de Fin de Máster.
- Crear los manuales de usuario e instalación para la reproducción del experimento.
- Ampliar mis conocimientos acerca de las distintas arquitecturas *Transformer*.
- Conocer distintas herramientas de extracción de características lingüísticas.
- Estudiar y conocer distintos objetivos del Procesamiento del Lenguaje Natural.
- Aumentar los conocimientos acerca del lenguaje de programación Python.
- Conocer y desarrollar la documentación del trabajo en Latex.

2.3. Estructura de la memoria.

A lo largo de esta sección se describen brevemente cada uno de los capítulos contenidos en la memoria.

El primer capítulo, ya visto, llamado Resumen consiste en una sinopsis breve acerca del tema tratado en el trabajo y su desarrollo. Además, se incluye su traducción al inglés.

En este segundo capítulo, se introduce brevemente el contenido del trabajo, se definen los objetivos necesarios a cumplir y se describe la estructura de la memoria.

En el tercer capítulo, Antecedentes y Estado del Arte, se revisan trabajos previos que se relacionan con el trabajo actual y son relevantes para la comprensión de la solución propuesta.

El cuarto capítulo, Situación de partida, se describen los condicionantes de partida del desarrollo del trabajo y que afectan al mismo. En este caso, se describe la competición sobre la que se apoya la experimentación.

El quinto capítulo, Hipótesis y restricciones, se identifican y registran las hipótesis de partida y las restricciones que se han utilizado para la elaboración de la experimentación de este trabajo.

El sexto capítulo, Alcance, se definen los límites del trabajo. Delimita un marco en el que se incluirán todos los resultados.

El séptimo capítulo, Estudio de alternativas y viabilidad, se enumeran y describen las distintas alternativas que se han tenido en cuenta y los criterios utilizados para su elección o descarte.

El octavo capítulo, Descripción de la solución propuesta, se describe la propuesta realizada para llevar a cabo la integración de características lingüísticas en redes neuronales profundas.

El noveno capítulo, Material y métodos, se describe la metodología y todos los medios empleados para realizar el trabajo experimental.

El décimo capítulo, Tecnologías utilizadas, se describen las tecnologías utilizadas en el proyecto. En este capítulo se habla sobre los lenguajes de programación utilizados, el software, las bibliotecas o servicios de los que se ha hecho uso.

El undécimo capítulo, Metodología, se presenta la metodología de desarrollo empleada y se justifican las raciones de su elección.

El duodécimo capítulo, Planificación, se presenta la planificación general y temporal del proyecto. Para ello de detalla una Estructura de Desglose de Trabajo, los perfiles profesionales necesario en el proyecto y una planificación temporal detallada con un diagrama de Gantt.

El capítulo treceavo, Presupuesto, se determina y justifica el coste económico de la elaboración del trabajo teniendo en cuenta el coste de los recursos humanos, el coste de los recursos materiales y los costes indirectos.

El capítulo catorceavo, Normas y referencias, se identifican las normas de cualquier tipo que hayan sido aplicadas en la elaboración del trabajo. Se muestran las normas y leyes utilizadas así como métodos, herramientas y métricas establecidas.

El capítulo quinceavo, Diseño inicial, se especifican aspectos como la arquitectura de los sistemas y los modelos de diseño de cada experimento.

El capítulo dieciseisavo, Desarrollo, se especifica todo lo relativo al desarrollo del trabajo. Se muestra el desarrollo y trabajo realizado en cada una de las iteraciones correspondientes según la metodología.

El capítulo diecisieteavo, Experimentación, resultados y discusión, se presenta la experimentación realizada y una discusión sobre los resultados de los experimentos realizados.

El capítulo dieciochoavo, Modelo de negocio, se propone un modelo de negocio viable para este proyecto.

Finalmente, en el último capítulo, el número diecinueveavo, llamado Conclusiones y trabajo futuro, se comprobará si se han cumplido los objetivos propuestos, se expondrán unas conclusiones y se añadirán unas mejoras.

Adicionalmente, se incluyen un conjunto de anexos con información relevante en el proceso.

Capítulo 3

Antecedentes y Estado del Arte.

Este trabajo final de máster consiste en el desarrollo de un sistema de redes neuronales con integración de características lingüísticas. A lo largo de este capítulo se describirá en que se basa este sistema y las tecnologías ya existentes. Para ello, se describen distintos conceptos como tecnología del lenguaje y redes neuronales sobre las que se sustenta y desarrolla en sistema generado.

3.1. Procesamiento del Lenguaje Natural.

La abundante cantidad texto en lenguaje natural que se genera en el mundo supone un gran volumen de conocimiento que es cada vez más costoso y complicado de extraer y gestionar. Es por ello que el Procesamiento del Lenguaje Natural (PLN) nació para resolver dicho reto, entre otros.

El Procesamiento del Lenguaje Natural, también llamado Lingüística Computacional o Tecnología del Lenguaje, es un área específica de la informática, concretamente de la Inteligencia Artificial (IA), que trata de analizar y representar textos naturales en uno o varios niveles de análisis lingüístico con el fin de lograr un procesamiento del lenguaje similar al humano en una serie de tareas o aplicaciones (Liddy, 2001). En otras palabras, según se narra en (Bird et al., 2009), por "lenguaje natural" entendemos un lenguaje que se utiliza en la comunicación cotidiana entre los humanos. A diferencia de los lenguajes artificiales, como los lenguajes de programación y las fórmulas matemáticas, los lenguajes naturales han evolucionado al pasar de generación en generación, y son difíciles de definir con reglas explícitas. Por ello, nos basamos en el PLN para manipular el lenguaje natural en la informática.

Dentro del Procesamiento del Lenguaje Natural nos encontramos con distintos niveles de procesamiento, esto se debe a la necesidad de facilitar y, sobre todo, modularizar las tareas que engloba este proceso. Los niveles más comunes en los que se divide el Procesamiento del Lenguaje Natural son (Khurana et al., 2022):

- Fonológico: Trata la interpretación de los sonidos del habla a través de las palabras.
- Morfológico: Trata la naturaleza sobre la que se componen las palabras. Las palabras se componen de morfemas, las unidades más pequeñas de significado.
- Léxico: Trata de la comprensión del significado que tienen las palabras individualmente.
- Sintáctico: Se centra en el análisis de las palabras de una frase para descubrir la estructura gramatical.
- Semántico: Determina los posibles significados de una frase centrándose en las interacciones entre los significados a nivel de palabra. Este nivel, además, incluye la desambiguación de palabras con múltiples sentidos.
- Discurso: Se centra en las propiedades del texto como un todo que transmite el significado al establecer conexiones entre las oraciones que lo componen.
- Pragmático: Se refiere al uso intencionado del lenguaje y utiliza el contexto para su comprensión. Pretende explicar el significado adicional en los textos sin estar realmente codificado en ellos por lo que se requiere un alto nivel de conocimiento del mundo.

Por mencionar algunas de las aplicaciones principales del PLN, nos encontramos con la traducción automática, la extracción de información, la revisión de textos, resolución de ambigüedad léxica, el reconocimiento de entidades, el reconocimiento del habla, el reconocimiento óptico de caracteres, la recuperación de información, la generación de resúmenes, la simplificación del texto o la generación de respuestas a preguntas (Chopra et al., 2013; Kumar, 2013). En este trabajo, la principal aplicación de la que se va a hacer uso es en la categorización o clasificación del texto.

La clasificación de textos es una tarea del PLN donde sistemas alimentados por grandes cantidades de texto en lenguaje natural, determinan para cada documento una etiqueta u otra según se defina. Los sistemas de categorización de textos intentan simular las decisiones de categorización humanas. Generalmente la clasificación puede llevarse a cabo mediante tres tipos de sistemas: sistemas basados en reglas, sistemas basados en máquinas o sistemas híbridos (Neupane, 2020).

Para llevar a cabo una clasificación con unos resultados más precisos se han aplicado y seguido diferentes técnicas entre las que nos encontramos la integración de características lingüísticas. La mayor parte de las investigaciones llevadas a cabo con la integración de este tipo de datos han sido desarrolladas con el idioma inglés y han demostrado obtener unos mejores resultados en tareas concretas (Choudhary & Arora, 2021).

3.2. Ingeniería de características en PLN.

La ingeniería de características o *feature engineering* es una tarea que consiste en generar y seleccionar características valiosas de otras características o de los datos extraídos sin procesar para que hagan que los algoritmos de aprendizaje automático funcionen mejor (Markovitch & Rosenstein, 2002). En el contexto del Procesamiento del Lenguaje Natural, la ingeniería de características implica la creación de características a partir de datos de texto que puedan utilizarse en modelos de PLN para mejorar su rendimiento. Un ejemplo son estructuras como los *word embeddings* (Collobert & Weston, 2008), que son representaciones numéricas de palabras que capturan el significado de las palabras de una manera que pueda ser entendida por los algoritmos de aprendizaje automático.

Las características lingüísticas pueden ser extraídas de un texto aplicando distintos métodos y los resultados pueden ser características gramaticales de los datos, frecuencia de palabras, información sintáctica, etc. (Dong & Liu, 2018; Markovitch & Rosenstein, 2002).

Integración de características.

La integración de características lingüísticas en un sistema se refiere a la incorporación de información sobre el lenguaje en la arquitectura de dicho sistema. Esto puede incluir información sobre la estructura del lenguaje, como la gramática y el significado de las palabras, así como información sobre el contexto en el que se utilizan las palabras. El objetivo de esta integración es mejorar la capacidad del sistema para procesar y comprender el lenguaje natural.

En la actualidad, existen múltiples trabajos donde se utilizan e integran las características lingüísticas para distintos objetivos específicos en numerosos tipos de redes o sistemas como, por ejemplo, en la predicción de utilidad para reseñas (Krishnamoorthy, 2015; Z. Liu & Park, 2015), detección de noticias falsas (Bauskar et al., 2019; J. Zhang et al., 2020) o detección de lenguaje de odio (García-Díaz et al., 2022).

3.3. Aprendizaje profundo.

El aprendizaje profundo o *Deep Learning* (DL) es un concepto en auge situado como una rama del Aprendizaje Automático (AA) o *Machine Learning* (ML) y consiste en aprender una parte de conocimiento en forma de múltiples niveles de representación y abstracción para generar información de nivel superior a partir de la existente en nivel inferior (Bengio, 2009). Es decir, el aprendizaje profundo se basa en el uso de redes neuronales artificiales con un conjunto de capas de procesamiento. Estas redes se asemejan a la estructura de la corteza cerebral y permiten a las máquinas "aprender" a partir de datos sin necesidad de programación explícita (Shinde & Shah, 2018).

De forma breve, en el DL, las capas inferiores que se sitúan cercanas a la entrada de datos aprenden características simples de los datos, mientras que en las capas superiores nos encontramos elementos más complejos derivados de las características de las capas inferiores. Por ello, el aprendizaje profundo es adecuado para analizar y extraer conocimiento útil tanto de grandes cantidades de datos como de datos extraídos de diferentes fuentes (L. Zhang et al., 2018).

Redes *Tranformer*.

En el año 2017, Google presentó una arquitectura centrada en el PLN, concretamente en la tarea de traducción automática, y apoyada en una técnica ya existente llamada *Attention* (Bahdanau et al., 2014). Esta nueva arquitectura presentaba resultados sobresalientes teniendo en cuenta otros tipos de redes más complejas como las Redes Neuronales Convencionales (CNN) o las Redes Neuronales Recurrentes (RNN). Esta técnica de DL avanzada es conocida como *Transformer* o redes *Transformer* (Vaswani et al., 2017) y se basa únicamente en mecanismos de atención.

Los modelos *Transformer* han demostrado una mejora en un amplio ámbito de tareas relativas al lenguaje como la clasificación del texto, la traducción automática o la generación de respuestas a preguntas. Además, los avances de estas redes en el dominio del PLN han despertado un gran interés para adaptar estos modelos fuera de éste campo, siendo aplicable a otros sistemas basados en la IA.

Para comprender mejor la estructura de las redes *Transformer*, debemos saber que se compone de dos partes principales:

- Codificador: esta parte procesa el texto de entrada, busca partes importantes y crea un *embedding* para cada palabra en función de la relevancia para otras palabras en la oración. Con *embedding* se refiere a la representación de palabras con vectores de números incrustando información adicional (Mikolov et al., 2013).
- Decodificador: esta parte recoge la salida del codificador y convierte el *embedding* en una salida de texto de nuevo.

En la figura 3.1 se muestra gráficamente la arquitectura de las redes *Transformer*.

Sin embargo, el éxito de estos modelos viene de la autoatención. Esta permite capturar las dependencias "a largo plazo" entre los elementos de una secuencia, es decir, dada una secuencia de elementos, la autoatención estima la relevancia de un elemento para otros elementos (Vaswani et al., 2017).

Algunos de los modelos existentes basados en la estructura *Transformer* son: Transformer-XL (Dai et al., 2019), entrenamiento previo generativo (GPT) (Radford et al.,

2018), representaciones de codificador bidireccional (BERT) (Devlin et al., 2018), modelo de lenguaje multilingüe (XLM) (Conneau & Lample, 2019), enfoque BERT robustamente optimizado (RoBERTa) (Y. Liu et al., 2019) o uno versión más pequeña y rápida de BERT (DistilBERT) (Sanh et al., 2019a), entre otras muchas variantes.

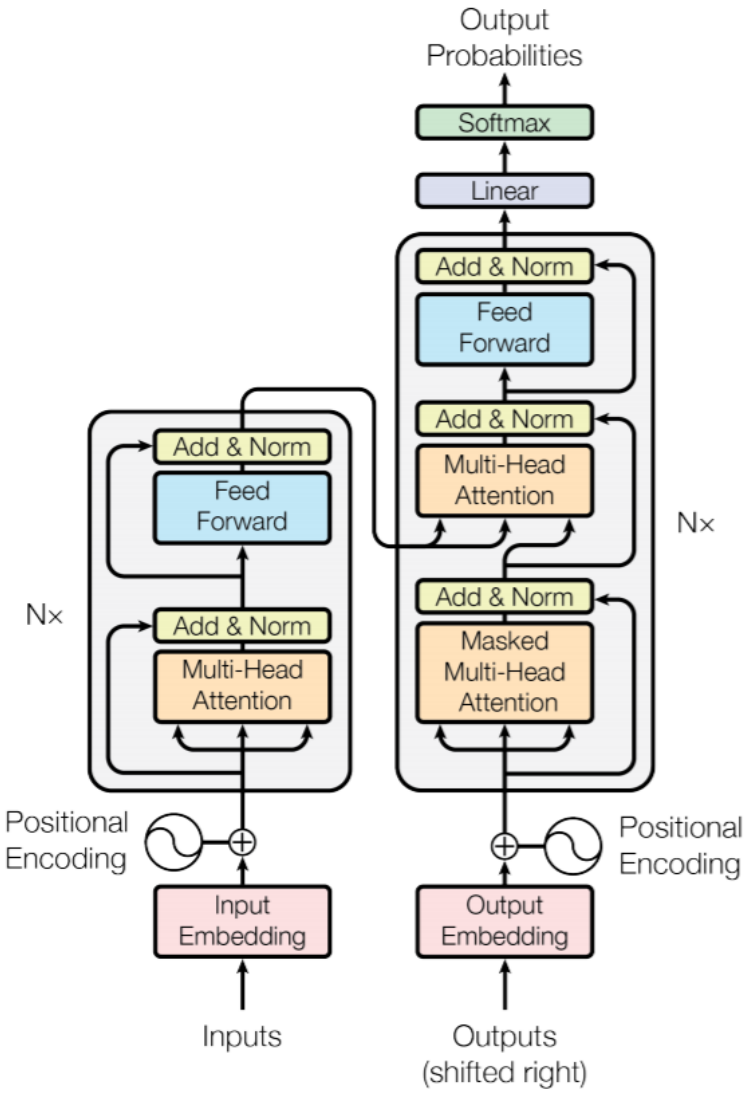

Figura 3.1: Arquitectura del modelo *Transformer*.

Capítulo 4

Descripción de la situación de partida.

A lo largo de este capítulo detallaré la situación inicial de la que se parte para el desarrollo del proyecto final de máster. Para ello, hablaré sobre el entorno actual, los datos de partida del experimento así como las deficiencias identificadas en dicha situación.

Descripción del entorno actual.

Para la realización del experimento hago uso del conjunto de datos facilitados en una competición. Una competición o campaña de evaluación es un conjunto de pruebas diseñadas para medir el rendimiento de un sistema, en este caso, en alguna tarea concreta del campo del PLN. Una campaña de evaluación puede incluir un conjunto de datos específico que se utiliza para evaluar el sistema junto con un conjunto de métricas para medir su rendimiento. El propósito es comparar distintos sistemas existentes.

En este caso los datos facilitados han sido extraidos de una competición contenida en el CLEF 2022[1] (*Conference and Labs of the Evaluation Forum*), concretamente de la tarea eRisk[2] 2022 Tarea 1: Detección temprana de signos de ludopatía. eRisk explora la metodología de evaluación, las métricas de efectividad y las aplicaciones prácticas de la detección temprana de riesgos en Internet. En general, esta competición contiene tres tareas asociadas cada una a distintos desordenes mentales:

- Tarea 1: Detección temprana de signos de juego patológico. El reto consiste en procesar secuencialmente un conjunto de publicaciones y detectar a la mayor brevedad rastros de juego patológico.

[1] https://clef2022.clef-initiative.eu/
[2] https://erisk.irlab.org/2022/index.html

- Tarea 2: Detección temprana de la depresión. El desafío consiste en procesar secuencialmente piezas de evidencia y detectar rastros tempranos de depresión lo antes posible.

- Tarea 3: Medir la gravedad de los signos de los trastornos alimentarios. La tarea consiste en estimar el nivel de características asociadas con un diagnóstico de trastornos alimentarios a partir de un hilo de envíos de usuarios.

En esta competición, que es una continuación de la tarea T1 de eRisk del año 2021[3], el reto principal consiste en la detección precoz del riesgo de juego patológico. Para ello se deben procesar secuencialmente envíos de los usuarios y detectar a la mayor brevedad rastros de juego patológico, también conocido como juego compulsivo o juego desordenado.

Esta tarea se ocupa principalmente de evaluar soluciones de Minería de Texto y, por lo tanto, se concentra en textos escritos en las redes sociales. Además, se indica que los textos deben procesarse en el orden en que fueron creados. De esta forma, se podrían aplicar sistemas que realicen efectivamente esta tarea para monitorear secuencialmente las interacciones de los usuarios en blogs, redes sociales u otro tipo de medios en línea.

La tarea de detección precoz de riesgos en Internet, concretamente aquella enfocada en la detección temprana de signos de ludopatía, facilita un conjunto de datos inicial diferenciado en dos tipos: datos de entrenamiento y datos de prueba.

La tarea se afronta desde dos perspectivas diferentes: como un problema de decisión binaria, y como un problema de clasificación (regresión).

- Como problema de decisión binaria, los mensajes tienen que ser etiquetados como positivos (etiqueta 1, es decir, adicción detectada) o negativos (etiqueta 0, no se detecta adicción). Cuanto antes detecte el sistema una adicción, este será mejor, como se refleja con la métrica *Early Risk Detection Error* (ERDE) y demás métricas propuestas por los organizadores y utilizadas para evaluar los sistemas, junto con las conocidas: precisión, cobertura y F1.

- Como problema de decisión de clasificación, en lugar de asignar etiquetas 0 o 1, se computa una puntuación de la estimación del riesgo de sufrir dicho trastorno. Se consideran diferentes métricas como las utilizadas para la recuperación de información para evaluar esta segunda visión de la tarea (P@10 o NDCG, entre otras).

Todas las métricas nombradas serán detalladas posteriormente.

[3]https://erisk.irlab.org/2021/index.html

Capítulo 5

Hipótesis y restricciones.

Este capítulo contiene las hipótesis y restricciones del proyecto. Se identifican las hipótesis de partida planteadas y las restricciones que se presentan para la elaboración de la experimentación del trabajo.

5.1. Hipótesis.

Tras la definición de los objetivos, en el primer capítulo, y el conocimiento de la situación de partida, capítulo 3, junto con el estado del arte de la cuestión, capitulo 2, al tratarse de un trabajo basado en experimentación, se deben plantear unas hipótesis de partida sobre las que se apoya todo el proceso de experimentación. Las principales hipótesis que se proponen son las siguientes:

I La integración de características lingüísticas en los modelos del lenguaje mejora los resultados en tareas de PLN.

II La integración de características lingüísticas en un sistema de clasificación o categorización de textos, en un modelo del lenguaje, ayuda a conseguir una mejor categorización de los mismos.

5.2. Restricciones.

Además de plantear los objetivos, igualmente es fundamental tener claras las limitaciones que forman parte de nuestro proyecto, como en cualquier otro. Estas limitaciones o restricciones pueden ser tanto de tipo intelectual, temporal o económico.

R 1. La principal restricción aplicable que nos encontramos es la limitación de la duración del trabajo. El Trabajo de Fin de Máster es definido como una asignatura con valor igual a 12

créditos académicos. Esto supone que la duración máxima y total del proyecto será de 300 horas (incluyendo todas las etapas del ciclo de vida).

R 2. Respecto a los recursos disponibles, para el desarrollo de este trabajo disponemos de un conjunto de máquinas o dispositivos limitados por lo que la capacidad de procesamiento de la que vamos a hacer uso debe tener correspondencia con la capacidad disponible para operar. En este caso disponemos de un ordenador portátil de uso personal y un cluster proporcionado por la universidad. Esto será mas detallado en capítulos siguientes.

R 3. Los conocimientos sobre los que se parte son básicos, estudiados en diversas asignaturas durante el Máster sobre IA. Ello ha conllevado un mayor tiempo de investigación de distintas arquitecturas para aplicar los conocimientos de la manera más correcta posible.

Capítulo 6

Alcance.

El objetivo de este capítulo es definir hasta dónde llega el trabajo. Dar un marco con el que delimitar todo lo que se incluye en los resultados.

- Dada la naturaleza del proyecto, los principales y únicos datos con los que se va a trabajar son los proporcionados por la competición descrita en el capítulo 4 por lo que no se va a trabajar con datos diferentes a éstos. No se van a usar nuevos datos para entrenar o evaluar modelos desarrollados y tampoco se van a utilizar un conjunto de datos con un idioma distinto al proporcionado en los datos iniciales.
- Además, se explorarán solamente los sistemas de integración de características en redes neuronales profundas diseñados y definidos en el capítulo 15 con los modelos de IA seleccionados en el capítulo 8, es decir, no se explorarán más sistemas o modelos de los definidos en este proyecto.

Para ello, a continuación se muestra una enumeración y contenido de todos los entregables del trabajo, tanto los resultados de los estudios teóricos y experimentales, como los que hacen referencia a la gestión y control de la ejecución del proyecto.

Entregables.

En esta sección se detallan los entregables que resultarán tras el desarrollo del estudio planteado en este documento. Los entregables principales que hacen referencia a la gestión y control del trabajo son:

- Memoria TFM: se desarrollará por completo la memoria del Trabajo de Fin de Máster que será entregada en formato PDF para mayor accesibilidad. Dicha memoria se refiere al documento actual.
- Código fuente en repositorio[1]: todo el código utilizado para el desarrollo de dicho trabajo

[1] https://gitlab.ujaen.es/ammr0032/trabajo-fin-master

será almacenado en un repositorio en la nube con el objetivo de poder ser visualizado y extraído de un modo sencillo y rápido. Dicho código se encontrará con los comentarios oportunos para su mejor comprensión. El código se compondrá de:

- Archivos referentes al análisis de los datos.
- Archivos referentes a la extracción de características.
- Archivos referentes a la experimentación y evaluación.

- Modelos: todos los modelos entrenados y evaluados, siguiendo los sistemas diseñados, tras la finalización de la experimentación de este proyecto serán almacenados en el repositorio de HuggingFace[2]. Esto incluye un total de doce modelos diferentes, uno por cada una de las experimentaciones propuestas. Los ficheros referentes a dichos modelos se almacenarán en el repositorio indicado.

  - Experimento 1.1. Nombre del modelo: ammr/RobertaLarge-Gambling1.1
  - Experimento 1.2. Nombre del modelo: ammr/RobertaLarge-Gambling1.2
  - Experimento 2.1. Nombre del modelo: ammr/RobertaLarge-Gambling2.1
  - Experimento 2.2. Nombre del modelo: ammr/RobertaLarge-Gambling2.2
  - Experimento 3.1. Nombre del modelo: ammr/RobertaLarge-Gambling3.1
  - Experimento 3.2. Nombre del modelo: ammr/RobertaLarge-Gambling3.2
  - Experimento 4.1. Nombre del modelo: ammr/RobertaLarge-Gambling4.1
  - Experimento 4.2. Nombre del modelo: ammr/RobertaLarge-Gambling4.2
  - Experimento 5.1. Nombre del modelo: ammr/RobertaLarge-Gambling5.1
  - Experimento 5.2. Nombre del modelo: ammr/RobertaLarge-Gambling5.2
  - Experimento 6.1. Nombre del modelo: ammr/RobertaLarge-Gambling6.1
  - Experimento 6.2. Nombre del modelo: ammr/RobertaLarge-Gambling6.2

[2]https://huggingface.co/

Capítulo 7

Estudio de alternativas y viabilidad.

Una vez establecidos los requisitos, alcance y objetivos del proyecto, se proceden a analizar las distintas formas de lograrlos ya que no hay un único camino para su resolución.

En el presente trabajo de fin de máster se ha planteado el desafío de integrar características lingüísticas en un sistema *transformer* para mejorar su capacidad para predecir si un usuario presenta rasgos de ludopatía o no. Para abordar este desafío, se ha tenido en cuenta una serie de alternativas, que se describen en este capítulo, junto con la justificación de la opción elegida, y las razones por las que las otras han sido descartadas. En el capítulo siguiente se presentará con mayor detalle la opción seleccionada.

7.1. Modelos de lenguaje.

Dentro de todas las posibilidades disponibles y tipos de redes neuronales que existen actualmente para crear modelos de Inteligencia Artificial, hoy día están en auge las redes *Transformers* por lo que se han evaluado distintos modelos para determinar cuál ofrece mejores resultados en la clasificación de textos.

Algunos de los modelos más conocidos y más valorados basados en la estructura *Transformer* son: representaciones de codificador bidireccional (BERT) (Devlin et al., 2018), un enfoque BERT robustamente optimizado (RoBERTa) (Y. Liu et al., 2019), una versión más pequeña y rápida de BERT (DistilBERT) (Sanh et al., 2019a), Transformer-XL (Dai et al., 2019) o entrenamiento previo generativo (GPT) (Radford et al., 2018). A continuación, se describen brevemente cada uno de ellos.

BERT.

El modelo *Bidirectional Encoder Representations from Transformers* (BERT) (Devlin et al., 2018), desarrollado en el año 2018 por Google, supera a los modelos del lenguaje

desarrollados en años anteriores porque es el primer sistema bidireccional no supervisado para el pre-entrenamiento del PLN. Sin supervisión significa que BERT se entrenó utilizando un corpus de texto sin formato y por bidireccional entendemos que un texto se procesa no solo de principio a fin sino también de fin a principio, lo cual ha sido fundamental para superar los límites de los modelos anteriores. Todo esto contribuye a que se pueda usar para ajustar tareas específicas de aprendizaje automático.

Otro de los puntos que diferencia a BERT respecto a otros modelos es el hecho de que tiene en cuenta el contexto para cada aparición de una palabra determinada, es decir, proporciona una representación diferente para una misma palabra dependiendo de su significado. Respecto a su estructura, BERT solo usa la parte del codificador de la red *Transformer* original. En la figura 7.1 vemos un ejemplo de su arquitectura cuando solamente tenemos una frase como entrada.

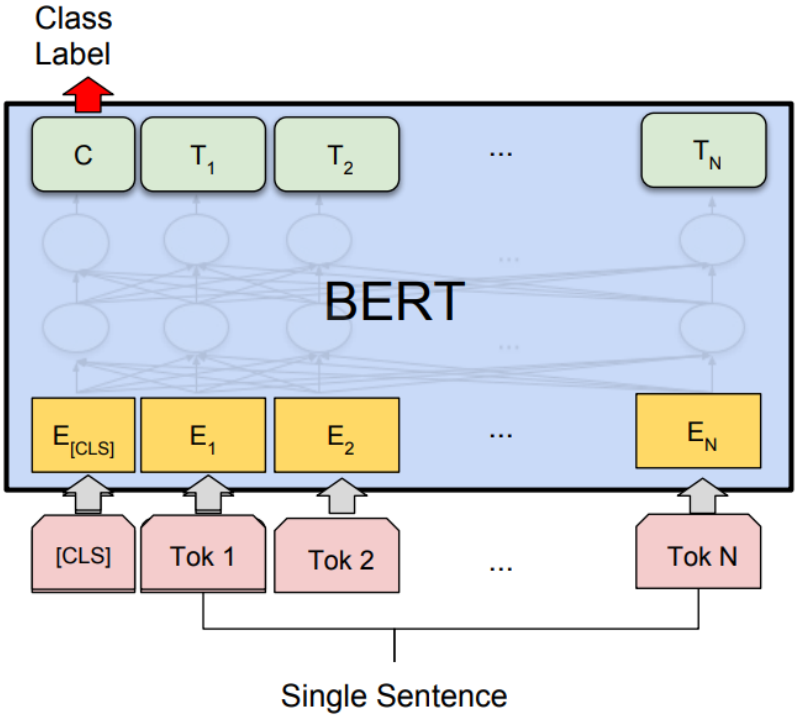

Figura 7.1: Ilustración del *Fine-tuning* de BERT en tareas de clasificación de una sola frase.

RoBERTa.

Respecto al modelo del lenguaje *Robustly Optimized BERT Approach* (RoBERTa) (Y. Liu et al., 2019) fue desarrollado en 2018 por Facebook AI Research como una versión más robusta y mejorada del modelo BERT. RoBERTa tiene una arquitectura similar a BERT con un procedimiento de preentrenamiento mejorado, modificando los hiperparámetros clave y

entrenando con más datos.

En varios estudios se ha comprobado que el modelo RoBERTa ha superado a BERT en diversos aspectos y proporciona mejores resultados que otros modelos desarrollados también como mejoras a BERT (Adoma et al., 2020; Murarka et al., 2020).

DistilBERT.

Como su propio nombre indica, BERT destilado (Sanh et al., 2019b), es una versión del modelo BERT, publicado en 2019, con un tamaño más pequeño, un 60 % más rápido, más barato de preentrenar y más ligero, reducido en un 40 %. Por lo que presenta muchas ventajas presentes en el modelo original junto con otras adicionales.

Transformer-XL

Los *Transformers* presentan la capacidad de aprender a largo plazo, sin embargo, están limitados por un contexto de longitud fija. Debido a dicha limitación, surge una nueva arquitectura llamada *Transformer-XL*, publicada en el año 2019 por Google AI. Esta arquitectura pretende tener la capacidad de atención más allá de un contexto de longitud fija sin interrumpir la coherencia temporal (Dai et al., 2019). Además, es un modelo autorregresivo, no bidireccional como BERT.

GPT.

Debido al gran numero de corpus sin etiquetar presentes en el ámbito del PLN, se propone una técnica semisupervisada que demuestra un rendimiento mejorado en una gran variedad de tareas como respuesta a preguntas, vinculación textual, clasificación de texto, etc. El modelo GPT, desarrollado por OpenAI en 2018, pretende eliminar en cierta medida los límites existentes por la cantidad pequeña de datos anotados para estas tareas específicas (Radford et al., 2018).

Recientemente, el modelo *Generative Pre-trained Transformer 3*, conocido como GPT-3, (Brown et al., 2020) ha demostrado ganancias sustanciales en muchas tareas y puntos de referencia del PLN.

Modelo elegido.

Dentro de todos los modelos existentes actualmente en el entorno del PLN, la arquitectura *Transformers* está en pleno auge por su gran capacidad de mejora demostrada en diversos estudios para varias tareas. Concretamente, centrándonos en un breve grupo de ellos como han sido descritos anteriormente, el modelo RoBERTa ha sido el más indicado para esta experimentación.

El modelo BERT ha sido la base para el nacimiento y crecimiento de un conjunto de modelos que mejoran todas las deficiencias y carencias encontradas en éste, por lo que partir de él, aunque podría resultar interesante, suponemos que no nos daría mejores resultados que otros modelos. En este caso no necesitamos un modelo más ligero, rápido o barato, y como consecuencia más limitado, debido al uso que vamos a darle por lo que DistilBERT también podría ser descartado. Por otro lado, los modelos RoBERTa y GPT están destinados a diferentes propósitos y son difícilmente comparables. RoBERTa está diseñado para tareas de clasificación de texto mientras que GPT está diseñado para tareas de generación de texto. Por tanto, RoBERTa es mejor para tareas de clasificación que Transformer-XL ya que es un modelo bidireccional. Por todos los motivos descritos, la arquitectura RoBERTa ha sido la elegida para el desarrollo de la experimentación propuesta.

7.2. Ingeniería de características.

Las características lingüísticas posibles son todas aquellas que puedan ser generadas a partir de los datos de partida. Se ha valorado incluir un número reducido de características lingüísticas, como por ejemplo las que se numeran a continuación:

- Volumetría: volumen de palabras según distintas categorías. Por ejemplo: número de palabras totales, número de adjetivos en el texto, etc.
- Diversidad léxica: valor de diversidad léxica presente. Por ejemplo: número de palabras distintas en el texto.
- Complejidad léxica: valor de complejidad léxica presente. Por ejemplo: número de palabras poco frecuentes utilizadas en el texto.
- Análisis de emociones: valor de distintas emociones presentes en el texto. Por ejemplo: valor presente de la emoción alegría o enfado en el texto, etc.
- Análisis de ofensividad: nivel de ofensividad presente en el texto. Por ejemplo: valor de ofensividad presente en el texto, persona o grupos a los que se dirige, etc.
- Recursos fonéticos: valores que ayudan a resaltar una parte concreta del texto por medio de sus sonidos. Por ejemplo: aliteración, uso de onomatopeyas, etc.

Todas ellas pueden ser consideradas para determinar cuáles son las más adecuadas para mejorar la precisión del sistema *transformer* en la predicción de rasgos de ludopatía.

Además, los datos de partida, vistos en el Capítulo 3, se presentan en forma de serie temporal donde cada texto viene asociado a una fecha y hora determinada, por lo que se podría tener en cuenta características temporales, como por ejemplo:

- Frecuencia de publicaciones por hora.
- Frecuencia de publicaciones en horario nocturno.
- Frecuencia de publicaciones por día de la semana.
- Frecuencia de publicaciones en fines de semana.
- Frecuencia de publicaciones por mes del año, etc.

Todas estas nuevas características podrían proporcionar más información no implícita en el texto para mejorar la precisión del sistema.

Características elegidas.

En este caso se ha asumido utilizar las cuatro primeras características lingüísticas (volumetría, diversidad, complejidad y análisis de sentimientos), teniendo en cuenta numerosos valores para cada una de ellas, para concluir si todo el conjunto, o una parte de él, alcanzan el resultado esperado en la experimentación. Esto es debido a que dichas características son las más utilizadas en el estado del arte actual y a que el análisis de ofensividad se apoya, en gran parte sobre el análisis de sentimientos. En este caso, los recursos fonéticos no se cree que sean una característica clave para el desarrollo del sistema propuesto.

Además, para tener en cuenta las características temporales habría que desarrollar sistemas adecuados a series temporales y desarrollar un análisis adecuado a este tipo de datos, que debido al tiempo disponible para la realización del trabajo de fin de máster no se han tenido en cuenta. Sin embargo, se consideran para un trabajo futuro.

Capítulo 8

Descripción de la solución propuesta.

Este capítulo contiene una descripción breve sobre la solución realizada y una enumeración de las características más significativas que permiten valorarla como la más idónea.

La solución propuesta en este trabajo de fin de máster consiste en integrar características lingüísticas en un sistema *transformer* para mejorar su capacidad para predecir si un usuario presenta rasgos de ludopatía o no. Para ello, se utilizará el modelo *transformer RoBERTa Large* y se incluirán las características lingüísticas de volumetría, diversidad léxica, métricas de complejidad y puntuaciones relacionadas con la emoción en una capa adicional introducida al final del sistema *transformer*. Esta solución se ha elegido como la más idónea por las siguientes características:

- RoBERTa es uno de los modelos transformer más avanzados y ha demostrado buenos resultados en tareas de clasificación de textos.
- La combinación de las características lingüísticas de volumetría, diversidad y complejidad léxica ha demostrado ser efectiva para mejorar la precisión de los sistemas de clasificación de textos. Y se desea analizar la efectividad añadida al introducir valores resultantes del análisis de sentimientos.
- La integración de estas características en una capa adicional al final del sistema *transformer* permite aprovechar las ventajas de ambos enfoques y mejorar la precisión del sistema en la predicción de rasgos de ludopatía.

Para afrontar el problema descrito se plantea una serie de experimentos en los que evaluar los resultados tras la integración de características lingüísticas en redes neuronales.

Para abordar esta tarea, se ha seguido un enfoque de aprendizaje supervisado. Para entrenar los modelos, se ha utilizado el conjunto de datos de entrenamiento proporcionado por los organizadores de eRisk (recordamos que consiste en una serie temporal de mensajes publicados por diferentes usuarios).

Debido a la limitada longitud de la secuencia de modelos de *Transformer* utilizados, en este caso un valor de 512 *tokens*, realizamos numerosas pruebas para elegir el número adecuado de *posts* que fueran más representativos de la patología (desde los más antiguos a los más recientes). Finalmente, se tomaron los 50 *posts* más recientes tras evaluar diferentes tamaños del modo en el que se detalla en la figura 8.1, para realizar los entrenamientos de los sistemas desarrollados. Teniendo en cuenta las publicaciones actuales de un usuario, se consideran únicamente las 50 últimas publicaciones realizadas a lo largo del tiempo. En el capítulo 17 se describe de forma más detallada la experimentación realizada.

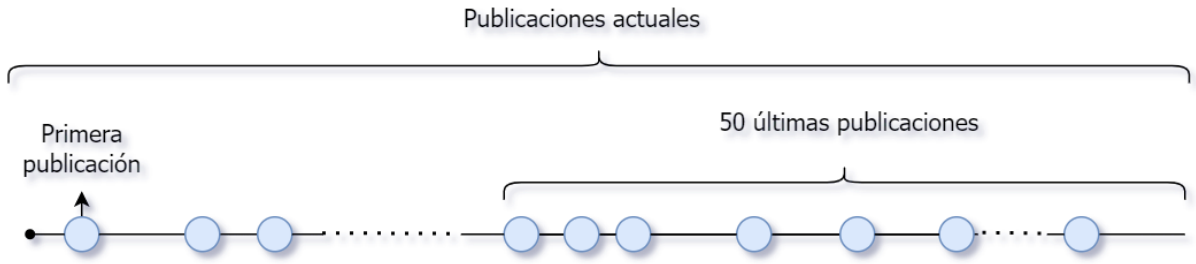

Figura 8.1: Diagrama de representación temporal de publicaciones realizadas por un usuario cualquiera.

Puesto que el objetivo principal de este trabajo es la integración de características lingüísticas en redes neuronales y su evaluación con respecto a la no evaluación, se plantean los siguientes experimentos detallados en la tabla 8.1.

Exp.	Contenido	Tipo Salida
Exp. 1.1	Embeddings	Regresión
Exp. 1.2	Embeddings	Clasificación
Exp. 2.1	Caracts. lingüísticas + Embeddings	Regresión
Exp. 2.2	Caracts. lingüísticas + Embeddings	Clasificación
Exp. 3.1	Caracts. lingüísticas (volumetría) + Embeddings	Regresión
Exp. 3.2	Caracts. lingüísticas (volumetría) + Embeddings	Clasificación
Exp. 4.1	Caracts. lingüísticas (diversidad) + Embeddings	Regresión
Exp. 4.2	Caracts. lingüísticas (diversidad) + Embeddings	Clasificación
Exp. 5.1	Caracts. lingüísticas (complejidad) + Embeddings	Regresión
Exp. 5.2	Caracts. lingüísticas (complejidad) + Embeddings	Clasificación
Exp. 6.1	Caracts. lingüísticas (sentimientos) + Embeddings	Regresión
Exp. 6.2	Caracts. lingüísticas (sentimientos) + Embeddings	Clasificación

Tabla 8.1: Principales experimentos planteados.

En el experimento 1 (exp. 1.1 y 1.2) se consideran solamente las características generadas por el modelo pre-entrenado de *Roberta Large (embeddings)*, en el resto de experimentos se tendrán en cuenta las características lingüísticas extraídas junto con los *embeddings*, integrándolas en las redes neuronales profundas según se describe en los capítulos siguientes esperando un mejor resultado. En el experimento 2 (exp. 2.1 y 2.2) se tiene en cuenta todos los tipos de características. En el experimento 3, 4, 5 y 6, se tienen en cuenta solo un tipo específico de estas (volumetría, diversidad léxica, complejidad léxica y análisis de sentimientos, respectivamente).

Además, el tipo de salida variará para cada uno de los experimentos descritos teniendo en cuenta diferentes tipos de salida: sistemas basados en regresión (exp. 1.1, 2.1, 3.1, 4.1, 5.1 y 6.1) y sistemas basados en clasificación (exp. 1.2, 2.2, 3.2, 4.2, 5.2 y 6.2). Recordemos que la clasificación es un proceso de encontrar una función que ayude a dividir el conjunto de datos en clases y la regresión es un proceso de encontrar las correlaciones entre las variables.

Capítulo 9

Material y métodos.

En este capítulo se tiene por objetivo el describir la metodología y los medios empleados para realizar el trabajo experimental. Por ello se incluye una descripción de los medios y los métodos empleados para realizar los estudios y las experimentaciones.

9.1. Material.

El material del que se dispone para el desarrollo de la experimentación consiste en un conjunto de datos, proporcionado por la campaña de evaluación introducida detallada en el Capítulo 3, y un sistema apoyado en un modelo de lenguaje pre-entrenado.

9.1.1. Modelo.

El modelo pre-entrenado utilizado, llamado *Roberta Large* está disponible de forma gratuita a través de la plataforma *Hugging face*[1]. *Hugging face* es una biblioteca de código abierto utilizada para crear modelos de aprendizaje automático.

Roberta Large es un modelo de lenguaje entrenado previamente en un gran corpus de datos en inglés de manera autosupervisada que sigue la arquitectura *transformer*. *Roberta Large* tiene un mayor número de capas e hiperparámetros que *Roberta Base* y proporciona mejores resultados que este, por lo que es elegido para el desarrollo del sistema. Por medio de este modelo, el sistema se apoya para clasificar a los usuarios proporcionados en los datos de la competición, tal y como se detallará en los siguientes capítulos.

[1] https://huggingface.co/

9.1.2. Conjunto de datos.

Para poder realizar la experimentación se necesitan unos datos sobre los que trabajar. Por ello se ha hecho uso de los datos proporcionados por la competición tomada como situación de partida.

Los datos facilitados por la competición, tanto los datos de entrenamiento, que son un total de 2.348 ficheros, como los datos de prueba, 2.079 ficheros, siguen el formato *eXtensible Markup Language (XML)*. En cada fichero se muestran cada una de las publicaciones que ha realizado un sujeto determinado indicando, entre etiquetas, distintos aspectos como el título, la fecha y hora de publicación, el contenido del mensaje, e información relativa a la procedencia de dicha publicación. Cabe destacar, que en todos los casos va a indicar procedencia de la red social Reddit.

Reddit[2] es una plataforma de código abierto donde los miembros de la comunidad pueden enviar contenido (publicaciones, comentarios o enlaces directos), votar envíos y las entradas de contenido están organizadas por áreas de interés (subreddits). Existen subreddits sobre diferentes condiciones médicas, como la anorexia, la ansiedad o la depresión. Además, los términos y condiciones de Reddit permiten utilizar su contenido con fines de investigación.

Respecto al lenguaje empleado, la mayoría de los textos están en el idioma inglés aunque hay una minoría de ellos en idioma español u otros idiomas. Esto es debido a que la extracción de los datos se llevó acabo desde una subcategoría de la red social Reddit (subreddit) en idioma inglés pero donde puede interactuar cualquier usuario independientemente del idioma.

9.1.3. Tratamiento de los datos.

Los datos facilitados por la competición, tanto los datos de entrenamiento como los datos de prueba, siguen el formato *eXtensible Markup Language* (XML). En cada fichero se muestran cada una de las publicaciones que ha realizado un sujeto determinado indicando, entre etiquetas, distintos aspectos siguiendo la estructura de ejemplo mostrada a continuación en la figura 9.1.

[2]https://www.reddit.com/

```
<INDIVIDUAL>
    <ID> subject1 </ID>
    <WRITING>
        <TITLE></TITLE>
        <DATE> 2017-08-28 20:59:39 </DATE>
        <TEXT> Yeah it is so dumb </TEXT>
        <INFO> Reddit post </INFO>
    </WRITING>
    <WRITING>
        ...
    </WRITING>
    ....
</INDIVIDUAL>
```

Figura 9.1: Muestra de ejemplo de formato de presentación de los datos iniciales.

Para un mejor manejo de los datos, paso la estructura dada en formato *XML* a una estructura en formato *CSV* más manejable. La nueva estructura presenta cinco columnas:

- Columna *index*: contiene el índice de cada *post* publicado por el sujeto.
- Columna *date*: contiene la fecha en formato AAA-MM-DD, de cada uno de los *posts* publicados por el sujeto.
- Columna *hour*: contiene la hora en formato HH:MM:SS, de cada uno de los *posts* publicados por el sujeto.
- Columna *title*: contiene el título de cada *post* publicado por el sujeto, si es que existe. Si no contiene *None*.
- Columna *text*: contiene el texto de cada *post* publicado por el sujeto, si es que existe. Si no contiene *None*.

9.1.4. Análisis de datos.

En cada fichero, se proporciona el id del sujeto correspondiente y un conjunto de publicaciones y comentarios realizados por este. A cada escrito le corresponde un título, una fecha, un contenido y un campo de información de procedencia. Respecto al lenguaje empleado, la mayoría de los textos están en el idioma inglés aunque hay una minoría de ellos en idioma español u otros idiomas.

Datos de entrenamiento.

Sabiendo que hay un sujeto por fichero, en total conocemos que hay 2.348 sujetos, de los cuales 164 son sujetos que presentan rasgos de ludopatía y 2.184 son sujetos que no presentan rasgos de ludopatía, es decir, son sujetos de control. Respecto a todos los sujetos, en general, tienen una media de 480'65 publicaciones con una desviación de 521'025, un mínimo de 10 publicaciones y un máximo de 2.001. Por otro lado, el número de palabras medio de un usuario es de 13.287'6 en total respecto a su historial, siendo la media más alta en los usuarios con rasgos de ludopatía. En la tabla 9.1 observamos un resumen de los datos de entrenamiento.

	Sujetos con patología	Control	Total
Num. sujetos	164	2.184	2.348
Num. envíos (posts y coments.)	54.674	1.073.883	1.128.557
Num. medio de envíos por sujeto	333'378	491'704	480'646
Desv. num. de envíos por sujeto	393'429	527'780	521'025
Num. max. de envíos por sujeto	1.356	2.001	2.001
Num. min. de envíos por sujeto	11	10	10
Num. medio palabras por sujeto	14.525'2	13.194'6	13.287'6

Tabla 9.1: Principales estadísticas de la colección de entrenamiento.

Datos de prueba.

Sabiendo que hay un sujeto por fichero, en total conocemos que hay 2.079 sujetos, de los cuales 81 son sujetos con rasgos de ludopatía y 1.998 son sujetos que no presentan rasgos de ludopatía, es decir, sujetos de control. Respecto a todos los sujetos, en general, tienen una media de 494'83 publicaciones con una desviación de 537'84, un mínimo de 2 publicaciones y un máximo de 2.001. En la tabla 9.2 observamos un resumen de los datos de prueba.

	Sujetos con patología	Control	Total
Num. sujetos	81	1.998	2.079
Num. envíos (posts y comentarios)	14.627	1.014.122	1.028.749
Num. medio envíos por sujeto	180'58	507'568	494,829
Desv. num. envíos por sujeto	247,32	542,57	537'84
Num. max. envíos por sujeto	1.341	2.001	2.001
Num. min. envíos por sujeto	10	2	2
Num. medio palabras por sujeto	7.754'17	15.837'6	15.522'6

Tabla 9.2: Principales estadísticas de la colección de prueba.

Se recuerda que estas estadísticas son desconocidas por los participantes en la campaña de evaluación eRisk, ya que el objetivo es hacer una evaluación del usuario publicación a publicación, en tiempo real, y detectar signos de riesgo lo antes posible.

9.1.5. Análisis gráfico de los datos.

Tras haber descrito las características de los conjuntos de texto utilizados, lo que incluye información como el tamaño de los conjuntos (es decir, cuántos textos hay en cada conjunto), el idioma en el que están escritos, y cualquier otra información relevante, en esta sección se analizan gráficamente los datos de partida para poder detectar patrones, tendencias, relaciones y estructuras latentes de los datos. Para ello se emplean distintos tipos de gráficos para una mejor comprensión, tales como: gráfico de cajas, gráfico de dispersión, mapa de calor, gráfico de tarta o nubes de palabras. El análisis que se ha llevado a cabo, de una forma tanto cuantitativa como cualitativa, ha sido extenso, por lo que a continuación se muestran los rasgos más significativos de los datos con los que se trabaja en este proyecto.

Volumetría.

Según se ha detallado en secciones anteriores de este capítulo, el volumen de los datos disponibles presenta una clara desproporción entre las clases presentes.

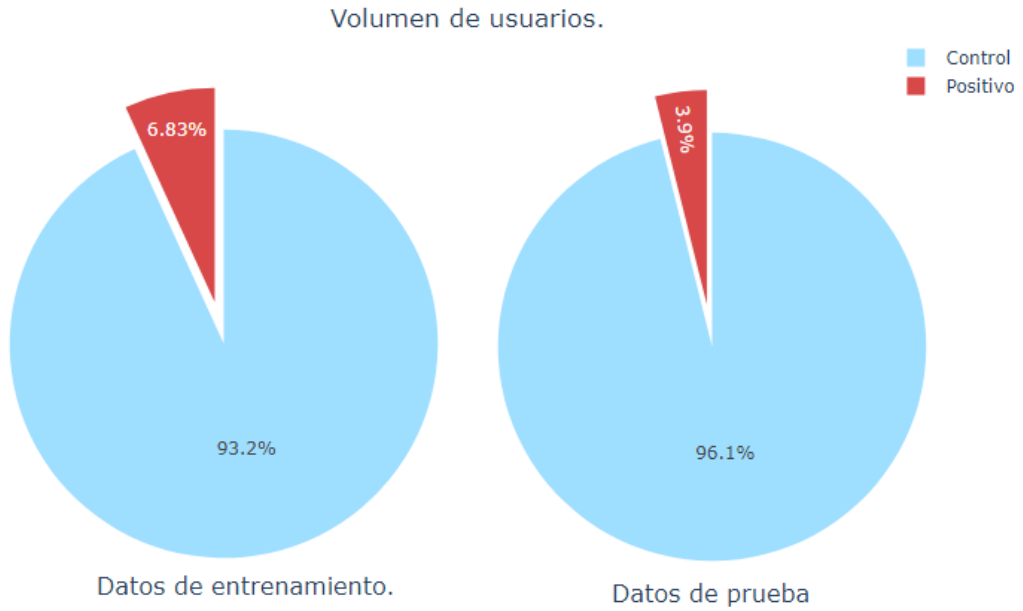

Figura 9.2: Representación en diagrama de tarta del conjunto de usuarios de entrenamiento y prueba y las clases presentes en ambos.

En la figura 9.2 podemos ver un esquema que muestra claramente el desbalanceo en los datos proporcionados tanto en el conjunto de entrenamiento como en el conjunto de prueba. En la imagen 9.3 se visualiza el desbalanceo presente no en el número de usuarios sino de publicaciones y escritos de cada grupo.

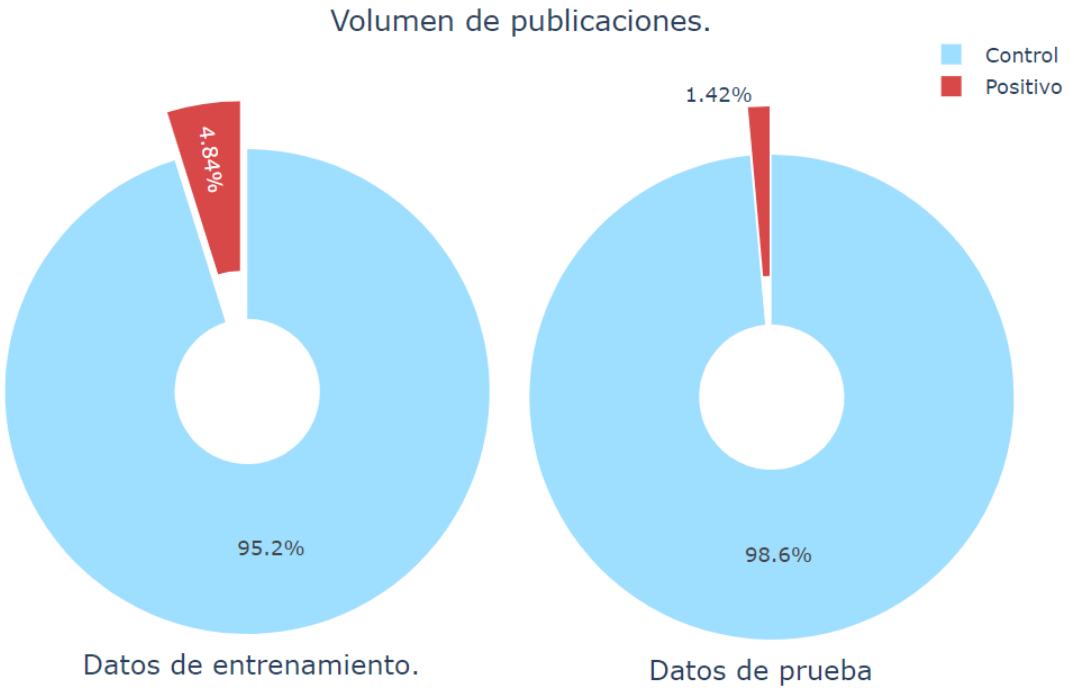

Figura 9.3: Representación en diagrama de tarta del conjunto de publicaciones de usuarios de entrenamiento y prueba y las clases presentes en ambos.

Frecuencia de palabras.

Esta sección muestra un análisis de las palabras más comunes en cada conjunto de texto. Esto podría ayudar a identificar las palabras más frecuentes en cada conjunto, y podría proporcionar información útil sobre las diferencias entre los conjuntos de texto de personas con rasgos ludópatas y el grupo de control. Por ello, se han extraído las palabras de cada conjunto de usuarios.

En la imagen 9.4 se visualizan las palabras más frecuentemente utilizados por el grupo de control, mientras que en la imagen 9.5 se visualizan aquellas más frecuentes para el grupo positivo en ludopatía.

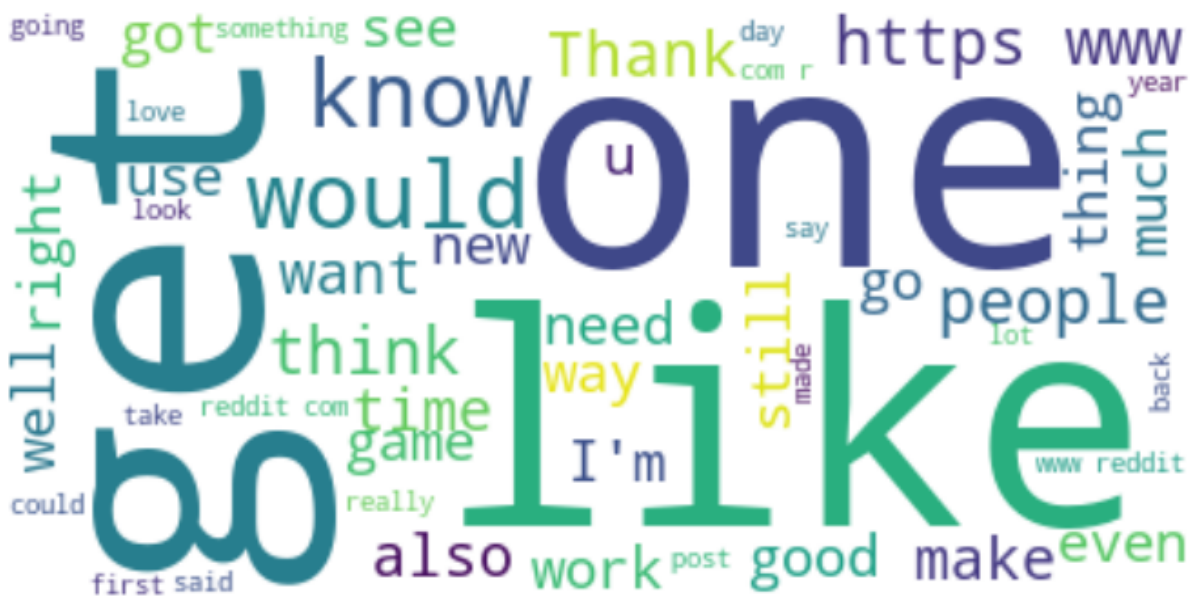

Figura 9.4: Nube de palabras más frecuentes en el grupo de control.

Palabras como *"gambling"*, *"win"*, *"money"*, *"casino"* o *"lost"* aparecen en la nube de palabras del grupo positivo mientras que en el grupo de control aparecen otras palabras como *"love"*, *"work"* o *"one"*.

Figura 9.5: Nube de palabras más frecuentes en el grupo positivo en rasgos de ludopatía.

Análisis de sentimientos.

También se considera la utilización de técnicas de Procesamiento del Lenguaje Natural, como el análisis de sentimientos o emociones, para obtener más información sobre los conjuntos de texto presentes. Esto podría ayudar a comprender mejor el contenido de los textos y a identificar patrones o tendencias en ellos que faciliten diferenciación de los conjuntos.

Por ejemplo, respecto al análisis realizado para el sentimiento de decepción existe una mayor presencia (una media mayor) en el conjunto de textos relativos a usuarios con rasgos de ludopatía frente a los que no, como se ve en la figura 9.6. Es decir, la mayor parte de los sujetos con rasgos de ludopatía presentan en sus publicaciones un nivel mayor de sentimiento de decepción que aquellos que no presentan rasgos de ludopatía.

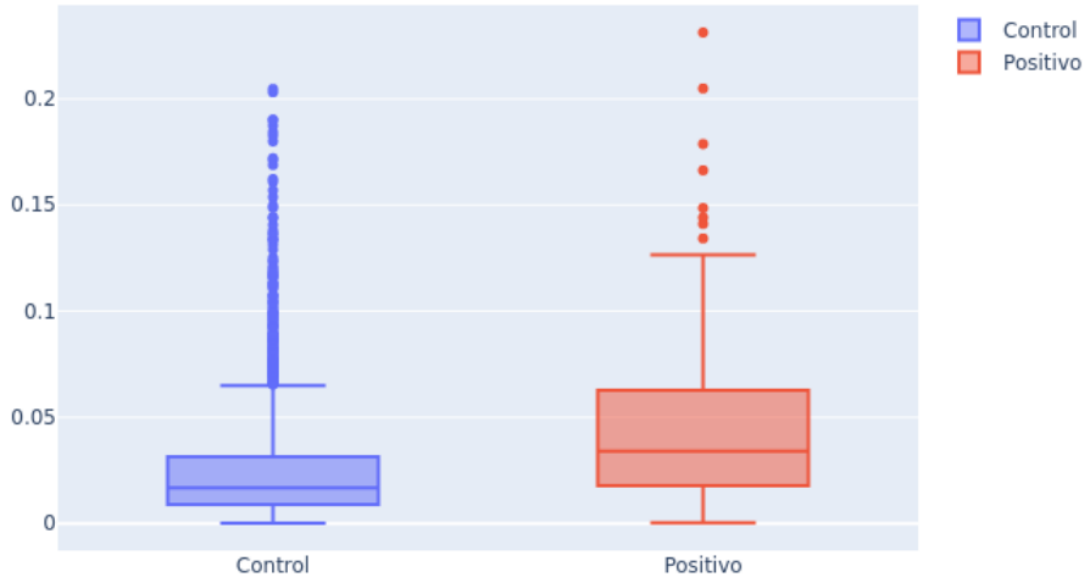

Figura 9.6: Representación en forma diagrama de caja y bigotes de la presencia del sentimiento de decepción en los textos de los usuarios relativos a ambas clases.

En la figura 9.7, se ve que el sentimiento de tristeza también está más presente en la mayor parte de los textos del grupo de sujetos positivo ya que los tonos son más claros (tienden a ser de color amarillo) en la mayor parte de ellos. Además, se puede ver la clara diferencia en el volumen de usuarios que hay para cada categoría, ya que en el grupo de control el número de sujetos es mucho mayor.

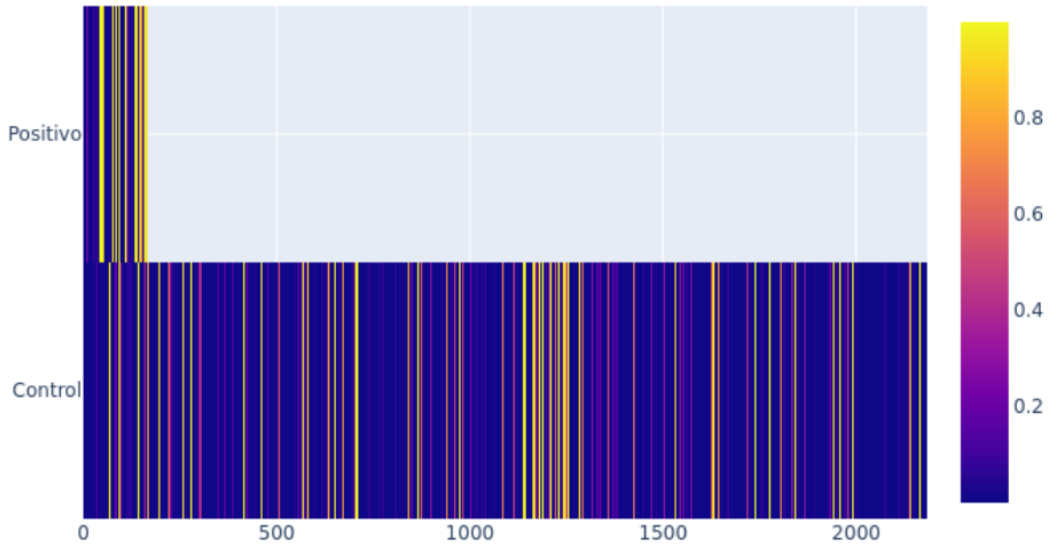

Figura 9.7: Representación en forma de mapa de calor de la presencia del sentimiento de tristeza en los textos de los usuarios relativos a ambas clases.

Deficiencias y carencias identificadas.

Como se ha podido observar en las descripciones dadas, los datos presentan un claro desbalanceo. El número de sujetos y, por tanto, de envíos realizados, es mucho mayor para aquellos que se sitúan en el grupo de control respecto de aquellos que se sitúan en el grupo de personas con riesgo de adición al juego. Esto es un problema frecuente en tareas con objetivos similares.

Además, el número de publicaciones totales es bastante alto (más de un millón de publicaciones) por lo que debemos tenerlo en cuenta a la hora de elegir la mejor solución para afrontar el problema. El tiempo de ejecución para algunas funciones puede verse incrementado por el gran volumen de datos.

Otro de los puntos a destacar, y que puede ser una carencia, es el hecho de que no sabemos a priori qué idiomas pueden aparecer en los textos, aunque, como ya se ha mencionado, la mayor parte están en inglés. Sin embargo, hay otros idiomas presentes, aunque en menor medida y que no han sido facilitados por los organizadores.

Por otro lado, muchas de las características lingüísticas analizadas previamente sobre los datos no parecen presentar una gran diferencia de forma visual entre los conjuntos, aunque un número de estas si permiten visualizar diferencias en los conjuntos.

9.1.6. Sistema de extracción de características.

Para realizar el análisis de los datos, se implementa un módulo capaz de extraer un número de características determinadas en puntos anteriores (características relativas a volumetría, a diversidad léxica, complejidad léxica y análisis de sentimientos). Estas características son analizadas para encontrar las diferencias más significativas entre los conjuntos de datos y, además, son utilizadas para su integración en un sistema de redes neuronales a implementar en la siguiente iteración.

9.2. Métodos.

En esta sección se describen los métodos utilizados a lo largo del desarrollo de este trabajo. Se ha tenido en cuenta distintos métodos como la validación cruzada para comprobar la robustez de los sistemas así como métodos de extracción de características en PLN.

9.2.1. Validación cruzada.

La validación cruzada o *Cross-validation* es un método de muestreo de datos para evaluar la capacidad de generalización de los modelos predictivos y evitar el sobreajuste (Refaeilzadeh et al., 2009). Hay varios tipos de validación cruzada. En este caso se hace uso del método *k-fold cross-validation*. En la validación cruzada de k-carpetas, el conjunto de aprendizaje disponible se divide, aleatoriamente y sin remplazo, en subconjuntos disjuntos de aproximadamente el mismo tamaño. El modelo se entrena utilizando k-1 subconjuntos que representan el conjunto de entrenamiento y con el subconjunto restante, que se denomina conjunto de validación, se mide el rendimiento. Este procedimiento se repite hasta que cada uno de los k subconjuntos haya servido como conjunto de validación. El valor final de rendimiento se calcula como la media de las medidas de rendimiento obtenidas en los k conjuntos de validación (Marcot & Hanea, 2021; Refaeilzadeh et al., 2009).

Es frecuente utilizar una validación cruzada de k veces usando $k = 5$ o $k = 10$, ya que se ha demostrado que estos valores producen valores más representativos (Marcot & Hanea, 2021). Se entiende que a mayor número de subconjuntos, mayor tiempo de ejecución será necesario. Por ello, en este trabajo se aplica *5-fold*, es decir, el conjunto de entrenamiento se divide en 5 conjuntos.

9.2.2. Extracción de características lingüísticas.

La parte fundamental en la que se basan los modelos de predicción desarrollados en este proyecto es la extracción de un vector de características de cada conjunto de mensajes por usuario proporcionado por la organización. Las características extraídas por nuestro sistema

recogen diferentes aspectos agrupados en diversas categorías: volumetría, diversidad léxica, métricas de complejidad y puntuaciones relacionadas con la emoción. Explicamos con más detalle en qué consisten estas características y los recursos utilizados para producirlas.

Para la generación de dicho vector de características, partimos del desarrollo de una clase que facilita varios métodos para el cálculo de diferentes métricas sobre el texto. Las métricas ofrecidas son: volumetría, diversidad léxica, complejidad léxica y análisis de emociones.

Volumetría.

En esta parte se extraen el número de palabras totales, el número de palabras únicas, el número total de caracteres, la longitud media de palabra, el número de lemas únicos, la longitud media por lema y el número de palabras de cada categoría lingüística para cada conjunto de mensajes por usuario. Las categorías lingüísticas principales de parte del discurso utilizadas son[3]: adjetivo (ADJ), adposición (ADP), adverbio (ADV), auxiliar (AUX), conjunción copulativa (CCONJ), determinante (DET), interjección (INTJ), nombre (NOUN), numeral (NUM), partícula (PART), pronombre (PRON), nombre propio (PROPN), puntuación (PUNCT), conjunción de subordinación (SCONJ), símbolo (SYM), verbo (VERB) y otro (X).

Diversidad léxica.

En esta parte se aplican diferentes técnicas para medir la diversidad léxica como las descritas a continuación[4]:

- *simple Type-Token Ratio (TTR)*: número total de palabras únicas (*types*) dividido por el número total de palabras (*tokens*).
- *root TTR*: número total de palabras únicas dividido por la raíz cuadrada del número total de palabras.
- *log TTR*: logaritmo en base 10 del número total de palabras únicas dividido por el logaritmo en base 10 del número total de palabras.
- *Maas TTR*: logaritmo en base 10 de los (*tokens*) menos logaritmo en base 10 de los (*types*) entre el logaritmo en base 10 de los (*tokens*) al cuadrado.
- *Mean Segmental Type–Token Ratio (MSTTR)*: esta estadística ayuda a comparar los valores de TTR de textos de distintos tamaños, ya que se utiliza el mismo tamaño de segmento, en este caso 50 tokens para cada texto. Por lo que MSTTR es el TTR medio de cada segmento no solapado de igual tamaño.

[3] https://universaldependencies.org/u/pos/all.html
[4] https://pypi.org/project/lexical-diversity/

- *Moving-Average Type–Token Ratio (MATTR)*: esta estadística también utiliza el tamaño de segmento detallado en el punto anterior. Por lo que MATTR es el TTR medio de todos los segmentos solapados posibles de igual tamaño.
- *Hypergeo-metric Distribution Diversity (HD-D)*: calcula, para cada tipo léxico de un texto, la probabilidad de encontrar alguno de sus *tokens* en una muestra aleatoria de 42 palabras extraídas del texto.
- *Measure of Textual Lexical Diversity (MTLD)*: se calcula como la longitud media de las cadenas de palabras secuenciales de un texto que tienen un valor TTR determinado. Cada palabra del texto se evalúa secuencialmente para su TTR.

En el trabajo de McCarthy y Jarvis (McCarthy & Jarvis, 2010) podemos conocer mejor acerca de estas métricas así como una evaluación de las mismas.

Complejidad léxica.

En esta parte hemos aplicado diferentes técnicas para medir la complejidad del texto. Las técnicas seleccionadas son las utilizadas en (López-Anguita et al., 2018). De forma resumida consisten en:

- Complejidad del léxico: basada en el hecho de que a más número de palabras diferentes en un texto, más dificultad hay para la comprensión.
- *Spaulding readability*: mide el vocabulario y la estructura de oraciones.
- *Sentence complexity*:mide el número de palabras por oración.
- *Automated readability index*: calcula el número medio de caracteres por palabra y del número medio de palabras por oración.
- *Height of the dependency tree*: mide la altura del árbol de dependencia.
- *Punctuation marks*: mide el promedio de signos de puntuación.
- *Fernández-Huerta's readability*: mide el promedio de sílabas por palabra y el promedio de palabras por oración que hay en el texto.
- *Flesch-Szigrist readability*: mide el número de sílabas por palabra y el número de palabras por oración que hay en el texto.
- *Gutierrez's comprehensibility*: mide el promedio de letras por palabra y el promedio de palabras por oración.

- *Readability*: mide el número de palabras, la media del número de letras por palabra y su varianza.
- *Minimum age of comprehension*: mide el promedio de sílabas por palabra y el promedio de palabras por oración.
- *SOL metric*: mide la legibilidad de un texto mediante el nivel de escolaridad necesarios para entender el texto.

Análisis de emociones.

Se han aplicado modelos de terceros, ya entrenados, para medir diferentes emociones que son expresadas en las publicaciones de los sujetos, como, por ejemplo, el enfado, la alegría o la tristeza. Estas emociones se obtienen usando dos modelos del lenguaje pre-entrenados disponibles en HuggingFace basados en la arquitectura de *Transformer*:

- DistilBERT[5]: es un modelo *DistilBERT* entrenado sobre un *dataset* (Saravia et al., 2018) que contiene textos anotados de acuerdo a 6 emociones extraídos de Twitter. Las emociones que evalúa son: amor, alegría, tristeza, enfado, sorpresa y miedo.
- BERT Emotion[6]: modelo *BERT* entrenado sobre un dataset (Demszky et al., 2020) que contiene textos anotados de acuerdo a 28 emociones extraídos de Reddit. Las emociones que evalúa son: amor, admiración, aprobación, alegría, atención, gratitud, neutral, optimismo, entusiasmo, deseo, curiosidad, realización, molestia, diversión, rechazo, confusión, tristeza, decepción, orgullo, enfado, sorpresa, repugnancia, alivio, remordimiento, vergüenza, miedo, dolor y nerviosismo.

[5]https://huggingface.co/bhadresh-savani/distilbert-base-uncased-emotion
[6]https://huggingface.co/bhadresh-savani/bert-base-go-emotion

Capítulo 10

Tecnologías utilizadas.

A lo largo de esta sección se detallarán las tecnologías utilizadas en el proyecto tales como lenguajes de programación, software, bibliotecas o servicios.

10.1. Software y servicios.

Para el desarrollo de este trabajo y consecución de los objetivos del mismo, se pueden tener en cuenta distintas tecnologías para afrontar el propósito del sistema planteado. Entre todas las opciones posibles, a continuación, se muestran aquellas más populares y económicas disponibles actualmente así como una descripción breve, ventajas y desventajas más destacadas.

Jupyter Notebook.

Jupyter Notebook[1] es una plataforma gratuita, de código abierto y basada en la web que permite a los usuarios crear y compartir documentos entre sí.

Figura 10.1: Logo de *Jupyter Notebook*.

Aunque presenta múltiples ventajas para distintas tareas, para el entrenamiento de modelos es difícilmente escalable y bastante lento, por lo que podría complicar el alcance de los objetivos.

[1] https://jupyter.org/

Google Colaboratory.

Google Colaboratory[2], también llamado Colab, es un producto de *Google Research*. Colab permite escribir y ejecutar código arbitrario de Python en el navegador. Es apropiado para utilizarlo en proyectos de aprendizaje automático, análisis de datos y educación. En términos técnicos, Colab es un servicio de *notebooks* de Jupyter alojados que no requiere instalación para usarlo y brinda acceso sin ningún costo a recursos computacionales, incluidas GPU.

Figura 10.2: Logo de *Google Colaboratory*.

Colab puede brindar recursos sin costo debido a que tiene límites de uso dinámicos que a veces fluctúan. No proporciona recursos garantizados ni ilimitados. Esto significa que los límites de uso generales, además de los períodos de tiempo de espera de inactividad, la vida útil de las VMs, los tipos de GPU disponibles y otros factores, varían con el tiempo.

La principal diferencia entre *Google Colab* y *Jupyter Notebook* es que el primero está basado en la nube y el segundo no. Esto significa que para trabajar en *Google Colab* no hay que descargar e instalar nada en el propio hardware y que el trabajo realizado se guardará y copiará automáticamente en la nube siendo accesible desde cualquier dispositivo. *Jupyter Notebook* se ejecuta en la máquina local y los archivos se guardan en el disco duro.

Clúster de supercomputación.

Otra de las alternativas es el uso de un clúster de supercomputación. Este clúster pertenece y está gestionado por el Centro de Estudios Avanzados en Tecnologías de la Información y la Comunicación (CEATIC)[3] de la Universidad de Jaén[4].

Es un clúster de supercomputación cuya finalidad es realizar tareas de Computación de Altas Prestaciones o *High Performance Computin* (HPC), utilizando gran cantidad de recursos de manera intensiva para la resolución de problemas y computación gráfica. En él se encuentran diversos nodos con una gran capacidad computacional con los que poder llevar a cabo los experimentos tratados en este proyecto.

[2]https://colab.research.google.com/
[3]https://www.ujaen.es/centros/ceatic/
[4]https://www.ujaen.es/

Visual Studio Code.

Visual Studio Code[5] es un editor de código fuente gratuito, de código abierto, disponible para múltiples lenguajes de programación y multiplataforma.

Figura 10.3: Logo de *Visual Studio Code*.

Alguna de las características de las que dispone son la incorporación de una terminal, la oportunidad de abrir múltiples proyectos a la vez, etc. Además, dispone de una gran variedad de extensiones descargables.

SmartGit

SmartGit es una herramienta software gratuita de control de versiones de código.

Figura 10.4: Logo de SmartGit.

El uso de esta herramienta es debido al apoyo que ofrece para mantener un orden y control de todos los cambios del proyecto. Permite mantener un orden y la recuperación de código en el caso debido. Además, ofrece la posibilidad de tener distintas versiones de un documento.

Software y servicios utilizados.

Para llevar a cabo la implementación del sistema deseado, se ha decidido hacer uso del clúster provisto por la Universidad de Jaén ya que serán necesarios una cantidad mayor de recursos para el entrenamiento de modelos de los disponibles por una computadora de uso personal o de los provistos por *Google Colaboratory* en su plan gratuito. Aunque también se hará uso de *Google Colaboratory* en su versión gratuita para pruebas de menor tamaño e impacto o realizar el análisis de datos.

[5] https://code.visualstudio.com/

Adicionalmente, *Visual Studio Code* será usado como entorno de desarrollo y como conexión al clúster de supercomputación mediante una de las extensiones disponibles que facilita la comunicación con el protocolo *Secure SHell* (SSH). En este caso se ha utilizado la extensión con nombre *Remote-SSH*[6] de Microsoft[7]. SmartGit se utiliza para subir distintos ficheros a repositorios y mantener un control de versiones.

Por tanto, mediante el clúster se utilizarán 2 Unidades de Procesamiento Gráfico (GPUs) NVIDIA V100 server para el entrenamiento de modelos de IA. En estas GPUs, cada Volta V100 tiene una memoria de 32GB, y respecto al número de núcleos que proporciona es de 5.120 núcleos CuDA FP32 y 640 núcleos Tensor.

10.2. Bibliotecas y lenguajes.

Para todas las tareas a realizar necesarias para la consecución del objetivo principal se hace uso de una serie de bibliotecas disponibles en el lenguaje de programación Python detallados a continuación.

Python

Python[8] es un lenguaje de programación orientado a objetos. Uno de los principales motivos del uso del lenguaje de programación Python es la gran variedad de bibliotecas disponibles para IA. Además, es un lenguaje muy sencillo de escribir y leer y facilita trabajar en el campo de *Big Data*, *Machine Learning* y *Data Science*, entre muchos otros.

Figura 10.5: Logo de *Python*.

Bibliotecas

Las bibliotecas utilizadas en el desarrollo de este trabajo de fin de máster abarcan todo tipo de tareas presentes en la planificación. Brevemente se numeran y describen cada una de las bibliotecas utilizadas (todas ellas disponibles para el lenguaje de programación Python):

- Lexical-diversity[9]: herramienta de cálculo de diversidad léxica. Se ha utilizado para calcular las características de diversidad léxica.

[6] https://github.com/Microsoft/vscode-remote-release
[7] https://www.microsoft.com/es-es
[8] https://www.python.org/
[9] https://pypi.org/project/lexical-diversity/

- Numpy[10]: herramienta de soporte para vectores de gran tamaño y funciones matemáticas. Se ha utilizado para aplicar sus funciones matemáticas.
- Math[11]: proporciona acceso a funciones matemáticas. Se ha utilizado para crear los métodos de evaluación de los modelos de inteligencia artificial.
- Pandas[12]: es una herramienta de manipulación y análisis de datos. Se ha utilizado para el manejo de los datos.
- Plotly[13]: crea gráficos interactivos de calidad. Se ha utilizado para crear sistemas gráficos en el análisis de datos.
- Pytorch[14]: biblioteca de aprendizaje automático. Permite trabajar con tensores, crear grafos de computación, calcular gradientes a partir de un error final y ajustar los tensores en base a dichos gradientes. Se ha utilizado para implementar los sistemas.
- Re[15]: proporciona operaciones de coincidencia de expresiones regulares. Se ha utilizado para la extracción de características lingüísticas.
- Scikit-learn[16]: herramienta para aprendizaje automático. Se ha utilizado para normalizar valores.
- Spacy[17]: herramienta diseñada para desarrollo de sistemas de lenguaje natural. Se ha utilizado para *tokenizar* el texto para extraer sus características lingüísticas.
- *Transformer*[18]: herramienta que provee modelos pre-entrenados para diferentes tareas. Se ha utilizado para la importación de modelos pre-entrenados en inglés.
- WordCloud[19]: herramienta de creación de nubes de palabras. Se ha utilizado para crear la nubes de palabras en el análisis de datos.
- XML [20]: interfaz para procesar archivos .xml. Se ha utilizado para la lectura y tratamiento de los datos iniciales que se encontraban en dicho formato.

[10]https://numpy.org/
[11]https://docs.python.org/3/library/math.html
[12]https://pandas.pydata.org/
[13]https://plotly.com/python/
[14]https://pytorch.org/
[15]https://docs.python.org/3/library/re.html
[16]https://scikit-learn.org/stable/
[17]https://spacy.io/
[18]https://pypi.org/project/transformers/
[19]https://pypi.org/project/wordcloud/
[20]https://docs.python.org/3/library/xml.html

Capítulo 11

Metodología.

Este capítulo presenta la metodología de desarrollo de software empleada y justifica las razones por las que se ha seleccionado.

Para comenzar una metodología de desarrollo (C. I. Rivas & Hernández, 2015; O. T. Gómez & Bacalla, 2010) permite desarrollar software a través de una serie de herramientas y métodos. Se ha de tener en cuenta los distintos tipos de metodologías y cuál es el que mejor encajaría con nuestro proyecto. Los dos enfoques entre los que se distribuyen los modelos existentes son las metodologías tradicionales y las metodologías ágiles.

Las metodologías tradicionales o clásicas (Hosting, s.f.) son útiles en proyectos controlados y de objetivos muy bien definidos al comienzo. Sin embargo, en la actualidad, son las metodologías ágiles (Hosting, s.f.; Marcos, 2001) las más utilizadas ya que permiten trabajar en un entorno cambiante y dinámico por lo que permiten una fácil adaptación.

Debido a que este proyecto tiene unas requisitos y especificaciones marcadas al inicio, las metodologías tradicionales parecen ser adecuadas para desarrollarlo. Los requisitos están especificados al tratarse de un proyecto académico, un trabajo de fin de máster, por lo que la opción de una metodología tradicional sería adecuada para la realización del proyecto. Además es una metodología apropiada para la realización de un proyecto de forma individual.

Metodología Incremental.

La metodología Incremental (Hosting, s.f.; Pérez, 2016) se basa en una estructura de desarrollo incremental, es decir, el desarrollo está dividido en pequeñas iteraciones o incrementos que están relacionados entre sí. En la figura 11.1 se puede ver el ciclo de vida de esta metodología.

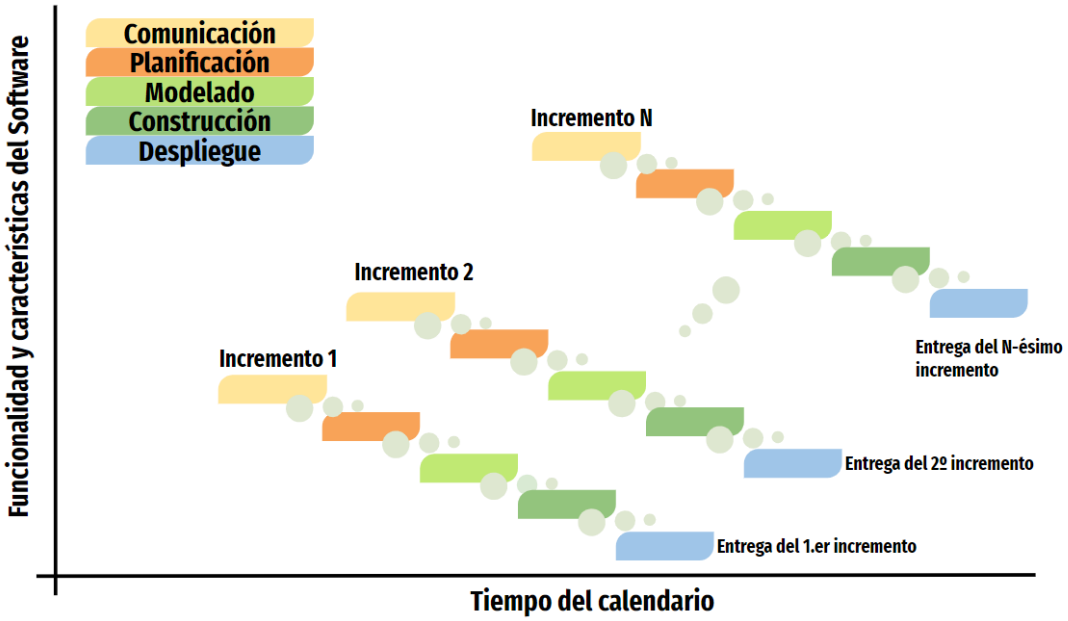

Figura 11.1: Esquema de metodología de desarrollo Incremental.

El modelo incremental o iterativo permite la división del trabajo y presenta la capacidad de poder hacer pequeñas mejoras o arreglos al incremento anterior. Hoy en día es aún una metodología muy utilizada. Algunas de las ventajas que presenta son:

- Presenta una estructura muy sencilla y fácil de llevar a cabo.
- Es ligeramente flexible y se adapta a los pequeños cambios.
- Reducción de riesgos debido a la división del trabajo en iteraciones.
- Posibilidad de ir mostrando entregas tempranas al cliente.

Las principales y preocupantes desventajas de este método están relacionadas con la falta de adaptación a los cambios, sin embargo, durante este proyecto los cambios serán mínimos debido a la alta definición de requisitos al inicio de este. Las fases principales que se llevan a cabo durante la metodología Incremental brevemente resumidas serían:

- Comunicación: se identifican los requisitos importantes y centrales del proyecto.
- Planificación: se identifican las tareas y se distribuyen en distintas iteraciones.
- Modelado: se definen los incrementos.
- Implementación: realización de los incrementos y refinamiento de funcionalidades.
- Revisión: validación de objetivos e incrementos.
- Lanzamiento: enviar entrega final y puesta en marcha.

Capítulo 12

Planificación.

Debido a que el presente proyecto es un Trabajo de Fin de Máster, no existen restricciones de tipo económico, sino de tipo temporal. Por consiguiente, los cálculos de tamaño del proyecto están condicionados por el tiempo disponible. En cuanto al esfuerzo, se dispone de tan un solo efectivo (la persona autora del trabajo de fin de máster).

A continuación, en las siguientes secciones, se detalla la estructura de desglose de trabajo, junto con todas las tareas establecidas para el periodo de tiempo marcado y su planificación, además de la estimación de recursos y costes.

12.1. Estructura de Desglose del Trabajo (EDT).

La Estructura de Desglose del Trabajo (EDT) o, en inglés, *Work Breakdown Structure* (WBS) consiste en la descomposición jerárquica de un trabajo o proyecto para definir el alcance total del proyecto (Devi & Reddy, 2012). El *Project Management Institute* define la Estructura de Desglose del Trabajo como la representación de la descomposición total de todo el trabajo que abarca el proyecto, de principio a fin.

En la imagen contenida en la figura 12.1 se puede ver una representación esquemática de la estructura de desglose de trabajo definida para este proyecto. En ella se pueden visualizar las tareas principales que determinan el trabajo a desarrollar en paquetes y actividades. Esta estructura es válida para cualquier tipo de proyecto, sin embargo, al tratarse de un proyecto teórico/experimental, en el cual se llevan a cabo experimentos para probar hipótesis, se incorporan puntos de decisión o conocimiento en el esquema en lugar de hitos o productos finales.

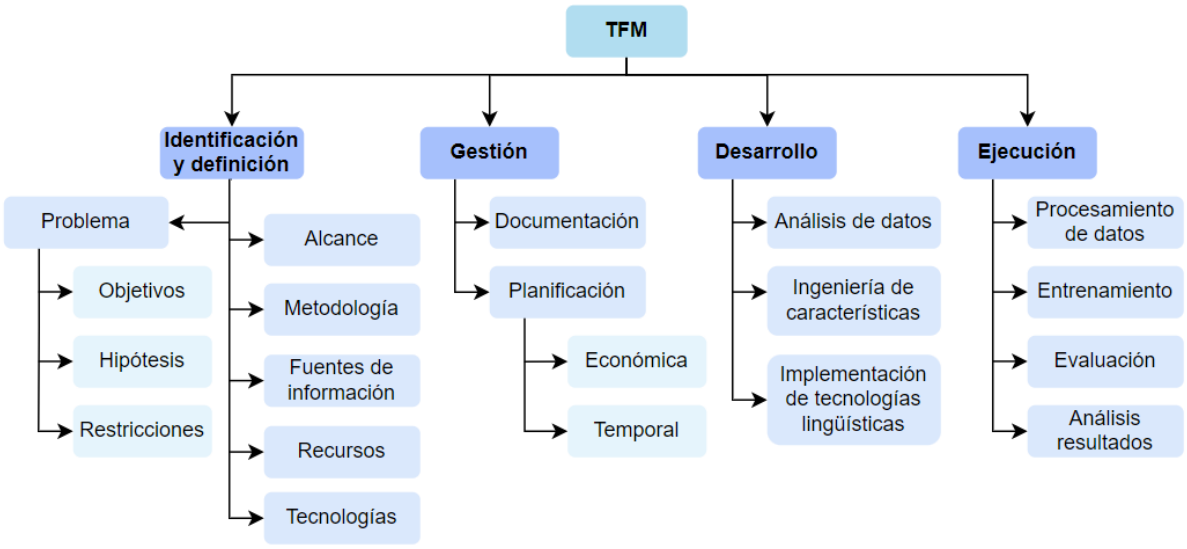

Figura 12.1: Estructura de Desglose del Trabajo.

En la figura 12.1 diferenciamos 4 niveles. El nivel superior (nivel 1) indica el título del proyecto. En el nivel de control (nivel 2) se marcan las fases principales del proyecto. En el nivel de paquetes de trabajo (nivel 3) se desglosan las fases en tareas más manejables. Por último, en el nivel de actividades (nivel 4) se incluyen tareas más pequeñas. En el Apéndice B encontramos el diccionario de la EDT.

12.2. Roles de trabajo.

En esta sección se plantean los roles o perfiles profesionales que se consideran necesarios, según los perfiles europeos de roles profesionales de las TIC (*The European ICT Professional Role Profiles*) que fueron publicados por el Comité Europeo de Normalización (CEN). Debido a la corta duración del proyecto, se consideran los roles de jefe de proyecto y de científico de datos.

- Jefe de proyecto: encargado de planificar, ejecutar y monitorizar las acciones del proyecto. Paquetes de la EDT en los que toma más importancia: problema, alcance, metodología, fuentes de información, recursos, tecnologías y planificación.
- Científico de datos: lleva adelante toda la experimentación y análisis de resultados. Paquetes de la EDT en los que toma más importancia: documentación, análisis de datos, ingeniería de características, implementación de tecnologías lingüísticas, procesamiento de datos, entrenamiento, evaluación y análisis de resultados.

12.3. Planificación temporal.

El objetivo de esta sección es presentar la planificación temporal del trabajo. Para ello se ha elaborado un cronograma que incluye los resultados parciales, objetivos intermedios y duración del trabajo a partir de la fecha de inicio del mismo.

A continuación, se estima la duración del trabajo de fin de máster desde el inicio del proyecto, estimado para el día 5 de septiembre, hasta su finalización, aproximadamente el día 2 de diciembre, considerando una media de 5 horas diarias y exceptuando los fines de semana y días festivos ocurridos durante el periodo estimado. En total la duración del proyecto, al tratarse de un Trabajo de Fin de Máster, es de 300 horas aproximadamente.

La planificación del proyecto respecto a la distribución de las tareas, de forma aproximada, sería la que se muestra en la siguiente tabla contenida en la figura 12.1.

Tareas	Horas aprox.	Fecha de inicio	Fecha de fin	Rol
Definición del problema	5	05/09/2022	09/09/2022	
Ident. de las fuentes de información	45	05/09/2022	30/09/202 2	Jefe
Definición del alcance	5	7/09/2022	09/09/2022	
Definición de metodologías	5	12/09/2022	14/09/2022	
Estimación de recursos	10	15/09/2022	23/09/2022	
Definición de tecnologías	15	15/09/2022	23/09/2022	
Planificación	10	26/09/2022	30/09/2022	
Análisis de datos	20	03/10/2022	07/11/2022	
Ingeniería de características	20	03/10/2022	11/10/2022	
Impl. de las tecnologías del lenguaje	80	03/10/2022	04/11/2022	
Procesamiento de los datos	10	26/10/2022	28/10/2022	Científico
Entrenamiento de los modelos	30	02/11/2022	16/11/2022	
Evaluación	10	10/11/2022	23/11/2022	
Interpretación de resultados	10	21/11/2022	30/11/2022	
Desarrollo de la documentación	25	03/10/2022	02/12/2022	
Resumen	300	05/09/2022	02/12/2022	

Tabla 12.1: Planificación temporal de las tareas previstas.

En general, se puede calcular la duración total del proyecto a priori en días cuyo valor sería de 60. Esto ha sido calculado, considerando 5 horas de trabajo diarias de media por lo que harían un total de 300 horas, 60 días * 5 horas, de trabajo. Las horas de trabajo aproximadas asociadas al rol de jefe de proyecto suman un total de 95 y las horas de trabajo aproximadas asociadas al rol de científico de datos suman 205, que en total equivalen a 300 horas de trabajo.

Otra forma de representación de la planificación es mediante un diagrama de Gantt. El diagrama de Gantt (Pastor, 2011) es una herramienta o sistema gráfico muy útil en el control y ejecución de actividades ya que en él se muestra una secuencia de tareas con una determinada duración distribuidas a lo largo del tiempo total establecido. En la figura 12.2 se presenta el diagrama del proyecto.

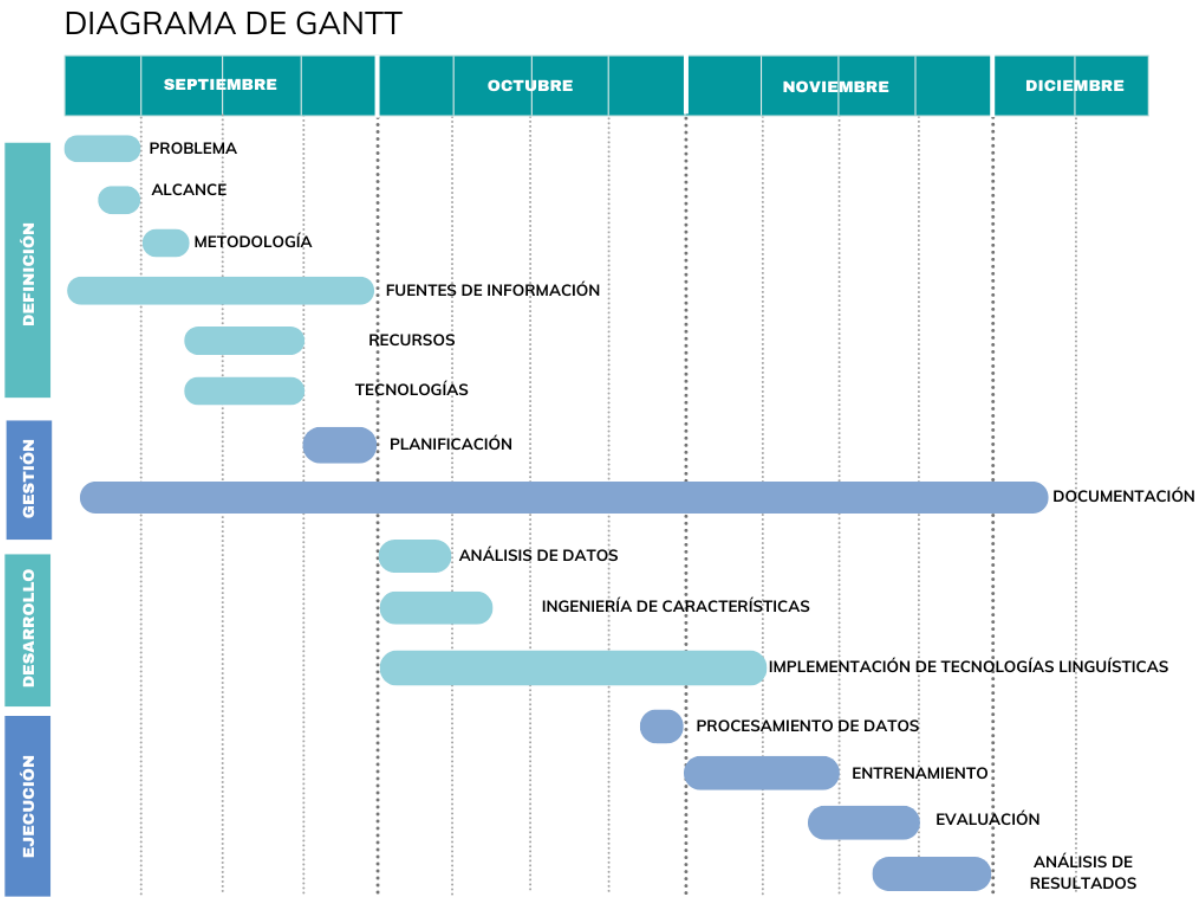

Figura 12.2: Diagrama de Gantt.

El diagrama de Gantt de la figura (Pastor, 2011) muestra las tareas establecidas en la EDT entre los meses de septiembre, octubre, noviembre y diciembre. Estos meses están divididos gráficamente cada uno de ellos en cuatro semanas para su mejor comprensión.

Capítulo 13

Presupuesto.

Aunque este sea un estudio técnico o un trabajo teórico/experimental, también tiene un coste asociado, ya que toda actividad, aunque sea de investigación, tiene un gasto de recursos. Por ello, el objetivo de este capítulo es detallar los costes dedicados a recursos humanos, recursos hardware y software y costes indirectos implicados en el proyecto.

13.1. Costes recursos humanos.

Respecto a los costes referentes a la categoría de recursos humanos se consideran los siguientes roles: jefe de proyecto y científico de datos.

Científico de datos

El sueldo base promedio de un científico de datos es de 35.000 € al año en España según indica la web del portal de empleo Glassdoor[1].

Sueldo al Mes (Científico de datos): 35.000 € / 12 Meses = 2.916,67 €

Considerando ese sueldo para 160 horas de trabajo mensuales en una jornada laboral normal en España (8 horas diarias), en este caso se han trabajado una media de 205 horas (sumando las horas referentes a los paquetes de trabajo destinados a este rol). Para esto calculamos el coste por hora.

*Coste por Hora (Científico de datos): (1 Hora * 2.916,67 €)/160 Horas = 18,23 €*

*Coste total (Científico de datos): 205 Horas * 18,23 € = 3.727,15 €*

[1]https://www.glassdoor.es/

Jefe de proyecto

Por otro lado, el sueldo base promedio de un jefe de proyecto es de 46.572 € al año en España según indica la web Glassdoor.

Sueldo al Mes (Jefe de proyecto): 46.572 € / 12 Meses = 3.881 €

Considerando ese sueldo para 160 horas de trabajo mensuales en una jornada laboral normal en España (8 horas diarias), en este caso se han trabajado una media de 95 horas (sumando las horas referentes a los paquetes de trabajo destinados a este rol). Por ello calculo el coste por hora.

*Coste por Hora (Jefe de proyecto): (1 Hora * 3.881 €)/160 Horas = 24,26 €*

*Coste total(Jefe de proyecto): 95 Horas * 24,26 € = 2.304,7 €*

Por tanto, los costes totales invertidos en recursos humanos para este proyecto (de una persona que concentra los dos roles profesionales) es de 6.031,85 €.

Coste total: 3.727,15 € + 2.304,7 € = 6.031,85 €

13.2. Costes recursos materiales.

En esta sección se estima un presupuesto de acuerdo a los componentes hardware y elementos software utilizados.

Recursos hardware

Respecto a los recursos hardware, para el desarrollo de este proyecto han sido necesarios los elementos de la tabla 13.1. En total suman 900€ de coste.

Concepto	Características	Coste
MSI Modern 14 B11SB-009XES	Procesador: 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80 GHz Memoria RAM: 16GB GPU: NVIDIA GeForce MX450 Sistema operativo: Windows 10	920 €
Nodos de cómputo	Procesador: 2x Intel(R) Xeon(R) Silver 4208 CPU @ 2.10GHz GPU: 2x NVIDIA Tesla V100 (32GB, Volta) Memoria RAM: 192 GB DDR4 @ 2933 Mhz Almacenamiento: 1.8 TB	19.598 €

Tabla 13.1: Recursos hardware.

Teniendo en cuenta el porcentaje de amortización vigente en España, accesible a través del portal de la sede de la agencia tributaria del Gobierno[2], a los equipos electrónicos se le aplica un 26 % de amortización por año. Además, el portátil se ha utilizado aproximadamente 3 meses, por lo que sería un 25 % de un año.

*Amortización (Portátil) = 920 € * 26 % = 239,2 € anuales * 0,25 = 59,8 € de gasto de amortización*

Para el nodo de computación, se ha utilizado aproximadamente 2 meses lo que equivale a un 16,67 % de un año.

*Amortización (Nodo Computación) = 19.598 € * 26 % = 5.095,58 € anuales * 0,16 = 815,27 € de gasto de amortización*

Recursos software

Respecto a los recursos Software de los que se ha hecho uso podemos visualizarlos en la tabla de la figura 13.2. En este caso el coste ha sido de 0€ debido a la elección de los recursos.

Concepto	Características	Coste
Google Colab	Herramienta web	0 €
Visual Studio Code	Editor de código fuente	0 €
Gitlab	Almacenarmiento de código	0 €
Bibliotecas	Accesible a través del lenguaje Python	0 €

Tabla 13.2: Recursos software.

Por tanto, el gasto total referente a recursos materiales empleados en el proyecto a sido de 875,07 €.

Coste total: 59,8 € + 815,27 € + 0 + 0 + 0 + 0 = 875,07 €

13.3. Costes indirectos.

Los cotes indirectos se entienden como los costes derivados del uso de instalaciones (alquileres, amortización de hipotecas, etc.), facturas de servicios (luz, agua, gas, teléfono, limpieza, vigilancia, etc.), gastos de gestión, hosting, conexión a Internet etc.

En la tabla 13.3 se estima el coste de los gatos indirectos para un mes de la localidad de Jaén, España.

[2]https://sede.agenciatributaria.gob.es/

Concepto	Coste estimado al mes
Luz y agua	150 €
Conexión Internet	30 €
Alquiler	400 €

Tabla 13.3: Costes indirectos.

Por lo que en total, la suma de gastos adicionales sería:

*Coste total (gastos indirectos) = (150 € + 30 € + 400 €) * 3 meses = 1.740 €*

13.4. Presupuesto total.

En resumen, si consideramos todos los gastos descritos en los puntos anteriores, el presupuesto del proyecto sería el presentado en la tabla de la figura 13.4.

Concepto	Coste estimado
Coste total recursos humanos	6.031,85 €
Coste total recursos materiales	875,07 €
Coste total recursos indirectos	1.740 €
Coste Total	8.646,92 €

Tabla 13.4: Presupuesto total del proyecto.

Por tanto, el coste total estimado para la realización de la experimentación de integración de características lingüísticas en redes neuronales es de 8.646,92€. Se extrae que el 69,76 % del presupuesto se aplica al personal trabajador.

Capítulo 14

Normas y referencias.

El objetivo de este capítulo es identificar la relación de normas, reglamentos, directrices y otros documentos de referencia de cualquier tipo que hayan sido tenidos en cuenta en la elaboración del trabajo.

14.1. Disposiciones legales y normas aplicadas.

En esta sección se indica el conjunto de disposiciones legales (leyes, reglamentos, ordenanzas, etc.) y las normas que son aplicables al trabajo.

Legislación

En este caso, la amortización calculada en el Capítulo 13.2 de este trabajo se ha realizado según la Tabla de amortización para contribuyentes acogidos al Método de Estimación Directa Simplificada del IRPF establecida en la Orden de 27 de marzo de 1998. Los elementos empleados en este trabajo se encuentran en el grupo 5 (Equipos para tratamiento de la información y sistemas y programas informáticos) de la tabla, por lo que se le aplica el coeficiente lineal máximo, un 26 %.

La legislación aplicada en este caso consta del Art. 30 Ley 35/2006 LIRPF. Normas para la determinación del rendimiento neto en estimación directa y de la Disposición Adicional 30ª Ley 35/2006 LIRPF. Libertad de amortización en elementos nuevos del activo material fijo.

Normas

Por otro lado, se ha seguido la norma CWA 16458-1:2018:*European ICT professionals role profiles* por el comité CEN/WS ICT para la selección de perfiles profesionales para los roles de trabajo elegidos contenidos en el Capítulo 12.2. Esta norma proporciona las descripciones completas de los perfiles europeos de funciones profesionales de las TIC.

14.2. Métodos, herramientas, modelos y métricas.

En esta sección se indicarán los métodos, métricas, programas, modelos y herramientas que se han utilizados en el desarrollo y estimaciones del trabajo (gestión de riesgos, costes, tiempos, etc.).

14.2.1. Herramientas.

Para la redacción de la documentación se ha hecho uso de Overleaf. Overleaf[1] es un editor colaborativo para el lenguaje de programación LaTeX basado en la nube que se utiliza, sobre todo, para escribir, editar y publicar documentos científicos.

Además, para la realización del diagrama de Gantt reproducido en el Capítulo 12.3 de este documento, se ha hecho uso de la herramienta Canva. Canva[2] es un software y sitio web de herramientas de diseño gráfico simplificado.

14.2.2. Métricas.

A continuación se describen todas las métricas utilizadas para la evaluación de los resultados generados tras la experimentación del trabajo. (Goutte & Gaussier, 2005) Para comenzar se tienen en cuenta cuatro conceptos fundamentales acerca de los valores predichos por los sistemas:

- Verdaderos positivos: se considera un valor como verdadero positivo o *True Positive* (TP) como aquel que se ha predicho como positivo, en una clase o categoría, y sabemos que realmente lo es, es decir, es el resultado cuando el modelo predice correctamente la clase positiva.

- Verdaderos negativos o *True Negative* (TN) es el resultado cuando el modelo predice correctamente la clase negativa.

- Falsos positivos o *False Positive* (FP) es el resultado cuando el modelo predice incorrectamente la clase positiva.

- Falsos negativos o *False Negative* (FN) es el resultado cuando el modelo predice incorrectamente la clase negativa.

[1] https://es.overleaf.com/
[2] https://www.canva.com/

Precisión

Con la métrica de precisión se mide la proporción de predicciones positivas del sistema que son válidas. En este trabajo, esta medida de clasificación estándar se aplica solamente a los verdaderos positivos.

$$Precision = \frac{TP}{TP + FP} \tag{14.1}$$

Precision at k

El valor de *Precision at k* o *P@k* consiste en el cálculo del número de valores predichos correctamente sobre el total. Donde *k* significa el número de publicaciones tomadas en cuenta para el cálculo.

$$P@k = \frac{valoresRelevantes}{valoresVistos} \tag{14.2}$$

Cobertura

Con la métrica de cobertura o *Recall* se conoce cuántos valores positivos son correctamente detectados por el modelo entrenado. En este trabajo, esta medida de clasificación estándar se aplica solamente a los verdaderos positivos.

$$Recall = \frac{TP}{TP + FN} \tag{14.3}$$

F1

F1 o *F-Score* es una métrica utilizada para medir el rendimiento de los modelos de aprendizaje automático de clasificación que combina los valores de precisión y cobertura. En este trabajo, esta medida de clasificación estándar se aplica solamente a los verdaderos positivos.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} = \frac{TP}{TP + \frac{1}{2} * (FP + FN)} \tag{14.4}$$

ERDE

ERDE (*Early Risk Detection Error*) (Losada & Crestani, 2016) es una medida de error que trata de introducir un valor de penalización por la tardanza al detectar verdaderos positivos. Tiene en cuenta los elementos siguientes:

- Una publicación de un usuario como k_u siendo $k_u \geq 1$.
- Un usuario como $u \in U$.
- Una decisión binaria para un usuario como $d_u \in 0, 1$
- El valor real de un usuario como $g_u \in 0, 1$
- El parámetro *o* controla el punto temporal en el que la penalización es mayor si no se detecta un positivo como verdadero.

Considerando una decisión binaria *d* tomada por el sistema en el punto *k*.

$$ERDE_o(d,k) = \begin{cases} c_{fp} & \text{if d=positive AND ground truth = negative (FP)} \\ c_{fn} & \text{if d=negative AND ground truth = positive (FN)} \\ l_{c_0}(k) * c_{tp} & \text{if d=positive AND ground truth = positive (TP)} \\ 0 & \text{if d=negative AND ground truth = negative (TP)} \end{cases} \tag{14.5}$$

Para este trabajo, se han utilizado los valores provistos por los organizadores de la competición descrita en la Situación de partida. En este caso, se tienen en cuenta los valores de $o = \{5, 50\}$, c_{tp}=1, c_{fn}=1 y c_{fp}=0,00389.

Latency$_{TP}$

La medida Latency$_{TP}$ se calcula sobre los verdaderos positivos detectados por el sistema y evalúa el retraso del sistema basándose en el número medio de escrituras que el sistema tuvo que procesar para detectar esos casos positivos.

$$latencyTP = median\{k_u : u \in U, d_u = g_u = 1\} \tag{14.6}$$

Speed

Esta métrica mida la velocidad a la que un verdadero positivo es detectado. El valor *speed* es igual a 1 para un sistema cuyos verdaderos positivos se detectan justo en la primera publicación. Para un sistema lento, que detecta verdaderos positivos después de cientos de publicaciones, se le asignará una puntuación de velocidad *speed* igual a 0.

$$latencyTP = 1 - median\{penalty(k_u) : u \in U, d_u = g_u = 1\} \tag{14.7}$$

$$penalty(k_u) = -1 + \frac{2}{1 + exp^{-p(k_u - 1)}} \tag{14.8}$$

Donde *p* es un parámetro que determina cómo de rápido el valor que penaliza debería incrementarse. En este caso $p = 0.0078$.

Latency-weighted F1

Esta métrica combina la eficacia de la decisión y el retraso de la decisión y es igual a la multiplicación entre F1 y *speed*.

$$F_{latency} = F * speed \tag{14.9}$$

NDCG

La Ganancia Acumulada Descontada Normalizada (NDCG) o *Normalized Discounted cumulative gain* es una medida que penaliza si un elemento relevante aparece bajo en el ranking. Suma los verdaderos positivos clasificados en el orden inducido por las puntuaciones predichas, después de aplicar un descuento logarítmico. Luego se divide por la mejor puntuación posible para obtener una puntuación entre 0 y 1.

$$NDCG = \frac{GananciaAcumuladaDescontada(DCG)}{GananciaAcumuladaIdealDescontada(IDCG)} \tag{14.10}$$

$$DCG = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2 (i + 1)} \tag{14.11}$$

$$iDCG = \sum_{i=1}^{REL_p} \frac{2^{rel_i} - 1}{\log_2 (i + 1)} \tag{14.12}$$

NDCG at k

El valor de *NDCG at k* o *NDCG@k* consiste en el cálculo de *NDCG* para los *k* primeros valores en una lista ordenada.

$$NDCG@k = \frac{DCG@k}{IDCG@k} \tag{14.13}$$

Capítulo 15

Diseño inicial.

En este capítulo se especifican aspectos como las arquitecturas de los sistemas propuestos y los modelos de diseño correspondientes.

15.1. Especificaciones del sistema.

El objetivo de esta sección es presentar la especificación detallada de los sistemas, incluidos los diagramas que fueran necesarios, que servirán de base para el análisis y diseño del sistema.

El principal motivo de este trabajo es la experimentación basada en la integración de características lingüísticas en redes neuronales profundas. Para ello hacen falta características lingüísticas y una red neuronal profunda donde integrarlas, además de otros sistemas donde comparar los resultados. Por ello se consideran dos tipos distintos de entradas de datos:

- *Embeddings* resultantes del paso del conjunto de publicaciones de cada usuario a través del modelo *RoBERTa Large*. En este proceso el texto en lenguaje natural se *tokeniza* de forma que es entendible por un sistema.
- Características lingüísticas resultantes de la extracción del conjunto de publicaciones de cada usuario. Estas características engloban un total de 4 ámbitos: volumetría, diversidad léxica, complejidad léxica y análisis de emociones.

Cualquiera de estas entradas será útil para entrenar un clasificador. El clasificador es un sistema que recibe una información de entrada e indica una salida que puede ser numérica o relativa a una clase. En este caso las salidas que se proponen son de dos tipos:

- Regresión: predice un valor de tipo numérico.
- Clasificación binaria: se considera un problema de clasificación binario, tratado como multiclase con dos clase posibles.

Finalmente, para poder converger los dos tipos de entrada (características lingüísticas y *embeddings*) se añade una capa oculta en el clasificador. El clasificador se compone de una *feed-forward neural network (FNN)* o red neuronal prealimentada que además contiene la función que determine el tipo de salida que se desea. En esta capa se concatenarán los dos tipos de entradas antes de la predicción.

Según la experimentación detallada para la solución del problema, se tienen en cuenta seis tipos de sistemas, experimentos o estructuras. Todos ellos tienen una entrada y una salida proveniente de un clasificador tal y como se muestra en la figura 15.1

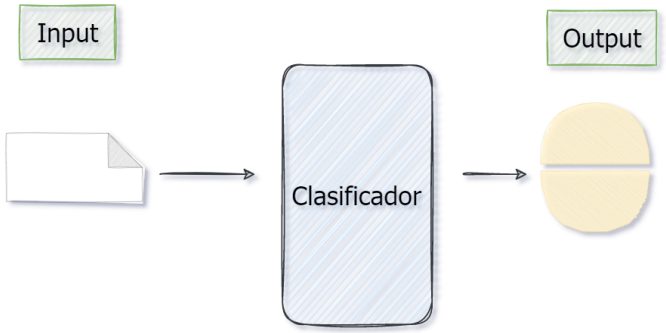

Figura 15.1: Arquitectura general de los experimentos. Un clasificador a partir de una entrada genera una salida.

La entrada del clasificador puede ser de dos tipos: las publicaciones de los usuarios como *embeddings*, las características lingüísticas extraídas de las publicaciones integradas con los *embeddings*. Esta entrada se pasa al clasificador para que determine una salida. Para ello se establece una *Feed-forward Neural Network (FNN)*, y se añade una capa oculta donde las neuronas toman un conjunto de entradas y producen una salida a través de una función de activación. La salida puede ser multiclase, si añadimos la función *softmax* o un valor de regresión si tomamos la salida sin ningún procesamiento.

A continuación se describen cada una de las arquitecturas propuestas para cada sistema.

Experimento 1.1

Según el experimento 1.1 se tiene en cuenta como entrada (*Input*) únicamente los *embeddings* de las publicaciones realizadas por cada usuario. Estos *embeddings* provienen de pasar las publicaciones del usuario por el modelo *RoBERTa Large*. Seguidamente se pasarán los resultados por un clasificador que dará como resultado un único valor como salida (*Output*). En este caso es un sistema de regresión.

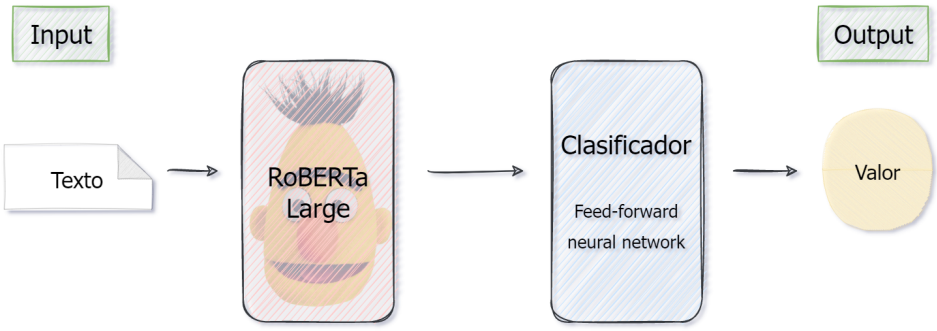

Figura 15.2: Arquitectura del experimento 1.1. El modelo recibe el texto *tokenizado* y el clasificador devuelve un único valor.

Experimento 1.2

El experimento 1.2 sigue el mismo esquema que el experimento 1.1, sin embargo, la salida no será un único valor. En este caso el sistema tiene una doble salida, es un sistema de clasificación.

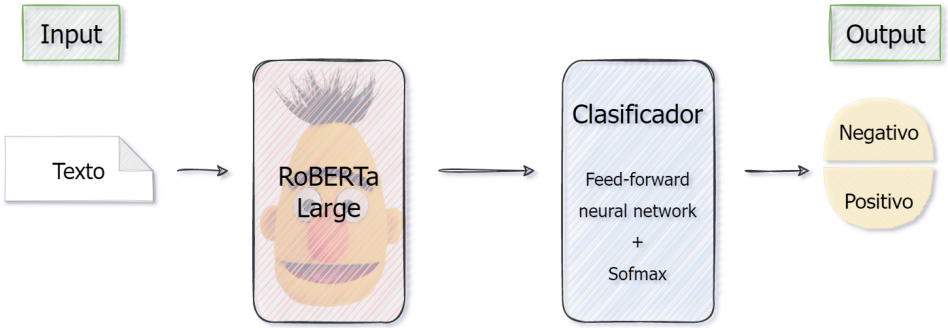

Figura 15.3: Arquitectura del experimento 1.2. El modelo recibe el texto *tokenizado* y el clasificador devuelve un valor para cada clase.

Experimento 2.1

Según el experimento 2.1 se tiene en cuenta como entrada (*Input*) los *embeddings* de las publicaciones realizadas por cada usuario. Estos *embeddings* provienen de pasar las publicaciones del usuario por el modelo *RoBERTa Large*. Seguidamente se pasarán los resultados junto con las características lingüísticas extraídas de las publicaciones por un clasificador que dará como resultado un único valor como salida (*Output*).

Es decir, los datos provinientes de la salida del modelo *RoBERTa Large* son concatenados a unas características lingüísticas de éstos generadas previamente y normalizadas entre los valores 0 y 1 para entrenar el clasificador. En este caso el sistema tiene una salida única, es un sistema de regresión.

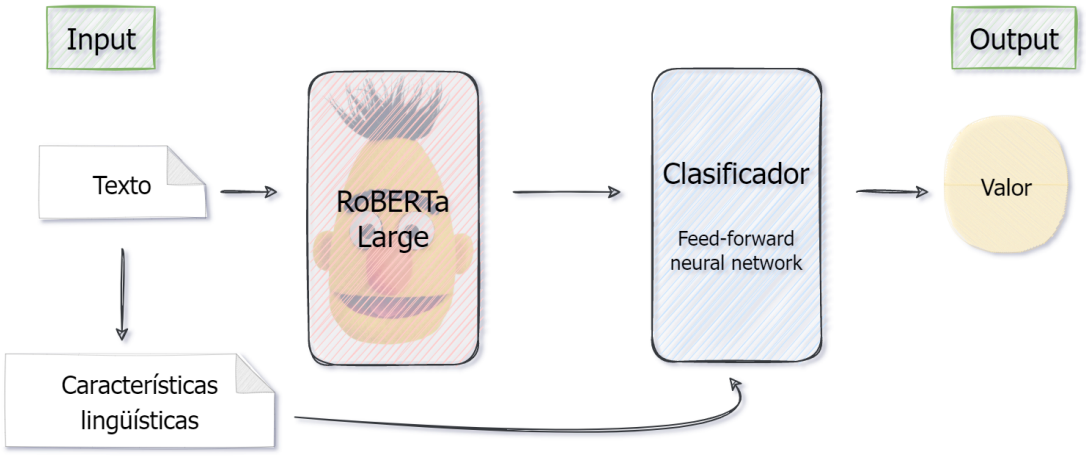

Figura 15.4: Arquitectura del experimento 2.1. El modelo recibe el texto *tokenizado* y el clasificador recibe las características lingüísticas de los textos y la salida del modelo y devuelve un único valor.

Este experimento sigue la misma estructura que los experimentos 3.1, 4.1, 5.1 y 6.1 , en los cuales en lugar de tener en cuenta todas las categorías de características lingüísticas solamente se tienen en cuenta una parte de ellas. Más adelante, en el capítulo siguiente, se detallará la estructura interna específica presente en este clasificador.

Experimento 2.2

El experimento 2.2 sigue el mismo esquema que el experimento 2.1, sin embargo, es un sistema de clasificación.

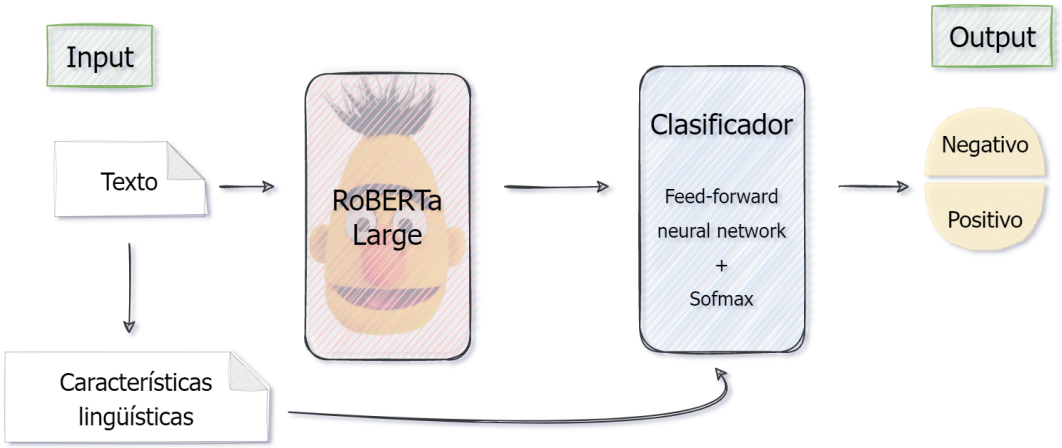

Figura 15.5: Arquitectura del experimento 2.2. El modelo recibe el texto *tokenizado* y el clasificador recibe las características lingüísticas de los textos y la salida del modelo y devuelve un valor para cada clase.

Este experimento sigue la misma estructura que los experimentos 3.2, 4.2, 5.2 y 6.2 , en los cuales en lugar de tener en cuenta todas las categorías de características lingüísticas solamente se tienen en cuenta una parte de ellas.

15.2. Análisis y diseño del sistema.

Esta sección tiene por objetivo presentar los documentos de análisis y diseño del sistema.

El sistema principal sobre el que se apoyan el resto, consiste primeramente en la concatenación de un número determinado de textos de un usuario. Una vez concatenados los textos, esa cadena se transforma en unas características ya sean *embeddings* o rasgos lingüísticos concatenados con estos. Finalmente, esos datos se pasan por el modelo que predice un resultado. En la figura 15.6 se puede ver de forma resumida el conjunto de pasos que engloban nuestros sistemas.

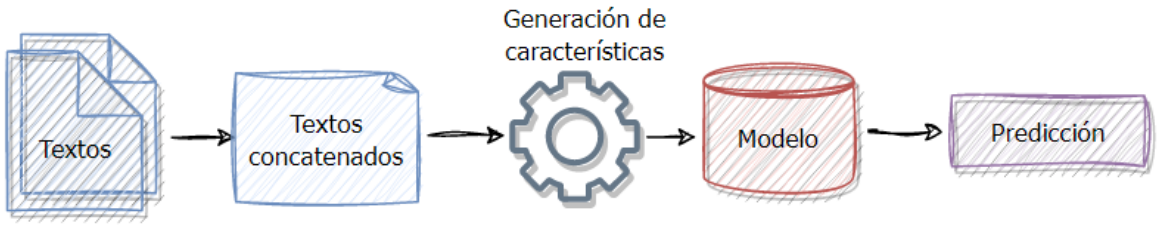

Figura 15.6: Arquitectura base de los sistemas a generar.

A continuación, basándonos en las especificaciones del sistema de la sección anterior, se analizan y diseñan los sistemas de una forma más concreta y detallada.

15.2.1. Experimento 1.1 y 1.2.

Como primer paso, el proceso que seguimos para predecir consiste en la extracción de los *posts*. Se concatenan los últimos N *posts* de cada usuario. Esta concatenación de *posts* es la entrada al modelo *RoBERTa Large*. A continuación, se aplica una red neuronal de alimentación directa (*Feed-forward Neural Network* (FNN)) con una capa oculta para generar las predicciones finales. La primera capa de la FNN final tiene una dimensión de entrada de 1024 (1024 del vector de incrustación generado por *RoBERTa Large*, solo se toma la *pooling output* que se calcula a partir del *embedding* correspondiente al *token* de inicio de secuencia.). La salida de la capa tiene una dimensión de 128. Estas salidas pasan por una capa *Drop Out* (DO) (con probabilidad 0,5) durante el aprendizaje y, por último, también se aplica una función de activación Unidad Lineal Rectificada (ReLU) antes de alimentar un último FNN con 128 entradas y 1 ó 2 salidas (las clases de decisión final, dependiendo de la configuración de la ejecución).

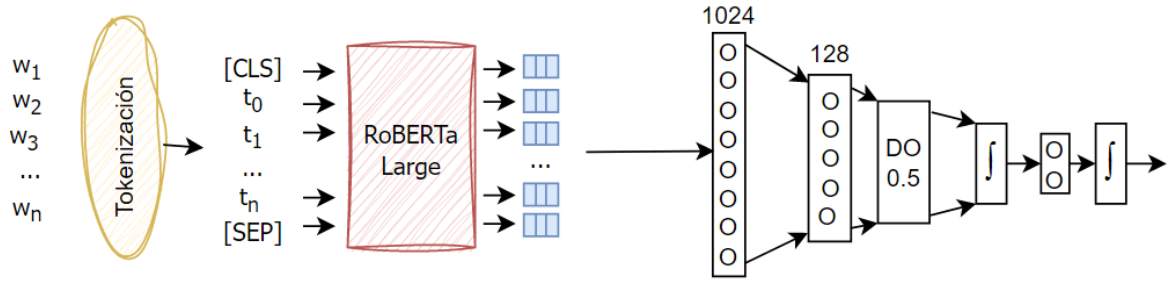

Figura 15.7: Arquitectura del experimento 1.2: decisión binaria.

La representación de la entrada es la tokenización del modelo *RoBERTa Large*. La diferencia para una tarea de regresión (experimento 1.1.) es la última FNN con una salida en lugar de dos salidas. A las características extraídas, como se ve en la figura 15.7 se le aplica una función *MinMaxScaler*, que pretende aplicar una normalización lineal a nivel de característica entre 0 y 1 donde 0 corresponderá con el valor mínimo encontrado en los datos y 1 con el valor máximo.

15.2.2. Experimento 2.1 y 2.2.

Como primer paso, el proceso que seguimos para predecir consiste en la extracción de los *posts*. Se concatenan los últimos N *posts* de cada usuario. Esta concatenación de *posts* es la entrada al modelo *RoBERTa Large*, un automodelo cuya salida se concatena con los vectores de características previamente generados y normalizados (de volumetría, complejidad, diversidad léxica y emociones). Una vez concatenados todos los rasgos, se aplica una red neuronal de alimentación directa (FNN) con una capa oculta para generar las predicciones finales. La primera capa de la FNN final tiene una dimensión de entrada de 1.103 (1.024 del vector de incrustación generado por *RoBERTa Large* y 79 características adicionales). La salida de la capa tiene una dimensión de 128. Estas salidas pasan por una capa DO (con probabilidad 0,5) durante el aprendizaje y, por último, también se aplica una función de activación ReLU antes de alimentar un último FNN con 128 entradas y 1 ó 2 salidas (las clases de decisión final, dependiendo de la configuración de la ejecución).

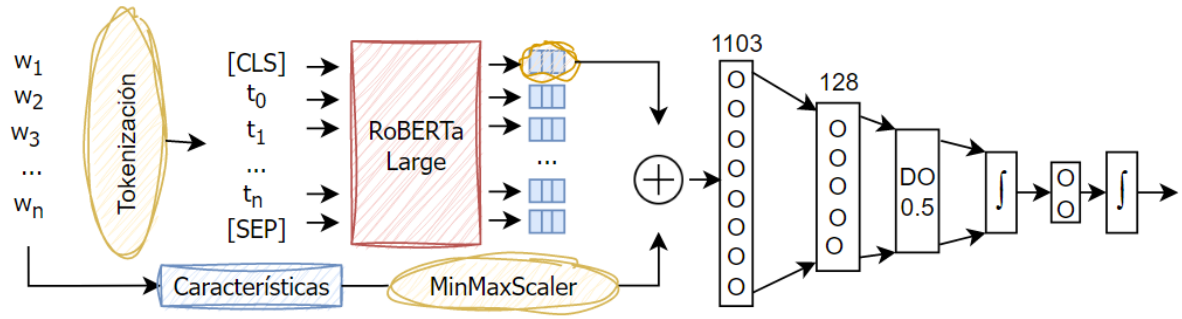

Figura 15.8: Arquitectura del experimento 2.2: decisión binaria.

En la figura 15.8 la representación de la entrada es la tokenización del modelo *RoBERTa Large*. Cada *embedding* de la salida del modelo es concatenado con su correspondiente vector de características lingüísticas normalizadas. La diferencia para una tarea de regresión (experimento 3.1.) es la última FNN con una salida en lugar de dos salidas.

15.2.3. Experimento 3.1 y 3.2.

Este proceso es prácticamente igual que el proceso seguido en el experimento anterior. La diferencia se encuentra en las características que se utilizan en el sistema, utilizando solamente las de la categoría de volumetría. En este caso, la primera capa de la FNN final tiene una dimensión de entrada de 1.048 (1.024 del vector de incrustación generado por *RoBERTa Large* y 24 características adicionales respecto a la volumetría) y la salida de la capa tiene una dimensión de 128.

15.2.4. Experimento 4.1 y 4.2.

Este proceso es prácticamente igual que el proceso seguido en el experimento 2. La diferencia se encuentra en las características que se utilizan en el sistema, utilizando solamente las de la categoría de diversidad léxica. En este caso, la primera capa de la FNN final tiene una dimensión de entrada de 1.032 (1.024 del vector de incrustación generado por *RoBERTa Large* y 8 características adicionales respecto a la diversidad léxica) y la salida de la capa tiene una dimensión de 128.

15.2.5. Experimento 5.1 y 5.2.

Este proceso es prácticamente igual que el proceso seguido en el experimento 2. La diferencia se encuentra en las características que se utilizan en el sistema, utilizando solamente las de la categoría de complejidad léxica. En este caso, la primera capa de la FNN final tiene una dimensión de entrada de 1.037 (1.024 del vector de incrustación generado por *RoBERTa*

Large y 13 características adicionales respecto a la diversidad léxica) y la salida de la capa tiene una dimensión de 128.

15.2.6. Experimento 6.1 y 6.2.

Este proceso es prácticamente igual que el proceso seguido en el experimento 2. La diferencia se encuentra en las características que se utilizan en el sistema, utilizando solamente las de la categoría de emociones. En este caso, la primera capa de la FNN final tiene una dimensión de entrada de 1.058 (1.024 del vector de incrustación generado por *RoBERTa Large* y 34 características adicionales respecto a la diversidad léxica) y la salida de la capa tiene una dimensión de 128.

Capítulo 16

Desarrollo.

El objetivo de este capítulo es especificar todo lo relativo al desarrollo (implementación) del trabajo. Como se ha seguido una metodología incremental, a continuación se detallan los incrementos que se proponen.

16.1. Primera iteración.

En esta primera iteración se propone definir el problema, principalmente. Para ello, esta fase se apoya en una amplia investigación.

Teniendo en cuenta la información disponible y encontrada durante la investigación, se busca determinar el alcance estimado del proyecto y la metodología a llevar a cabo (tras un análisis de las metodologías posibles). Además, se empieza a documentar aspectos teóricos relativos a este punto. En esta fase, se investiga sobre posibles soluciones al problema, problemas similares y sus resoluciones.

Finalmente, se decide seguir la metodología incremental y se estima un alcance realista.

16.2. Segunda iteración.

En esta segunda iteración se propone una búsqueda de recursos y tecnologías para la resolución del problema.

En este punto, continúa la investigación sobre las soluciones que se tomarán en este trabajo además de la búsqueda de recursos y tecnologías a utilizar. Analizando soluciones similares a problemas similares y soluciones que afectan positivamente a otro tipo de problemas. Además, se analizan las tecnologías disponibles y su adecuación a este proyecto así como los recursos útiles sobre los que llevar a cabo la experimentación. En esta fase, se analizan las características que podrían ser tomadas de los datos.

Finalmente, se decide la solución al problema, los recursos y las tecnologías a utilizar y se continua documentando toda la información generada.

16.3. Tercera iteración.

En esta tercera iteración se plantea realizar un análisis de los datos de partida extraídos como parte de los recursos, a la vez que se extraen distintas características lingüísticas de estos a partir del análisis realizado. Para ello, se implementan sistemas de extracción de características y análisis de datos que se tomarán en cuenta en las iteraciones posteriores. Además, toda la información extraída de los datos será documentada.

16.4. Cuarta iteración.

En esta iteración, se implementan los sistemas de entrenamiento y evaluación. Es decir, se implementan los sistemas de redes neuronales diseñados previamente, en los cuales se llevará a cabo la integración de características lingüísticas. Además, al mismo tiempo se desarrollarán los sistemas de evaluación correspondientes para ir probando los sistemas.

A la vez que se van implementando los sistemas, se van evaluando los datos y se va documentando todo el proceso.

16.5. Quinta iteración.

En esta quinta iteración se analizan los resultados de los sistemas generados en la iteración anterior. En este punto del proyecto, se comparan los resultados entre ellos teniendo en cuenta distintos tipos de métricas y evaluaciones. Además, se finaliza la documentación relativa al proyecto, donde se discuten los resultados obtenidos junto con las conclusiones extraídas de todo el proceso y el trabajo futuro propuesto.

Capítulo 17

Experimentación, resultados y discusión.

El objetivo de este capítulo es detallar la experimentación realizada, enumerar los resultados más relevantes y presentar una discusión sobre los mismos.

17.1. Experimentaciones y pruebas.

En esta sección se detallarán las experimentaciones y pruebas realizadas con todos los parámetros y características relativas a éstas.

17.1.1. Entrenamiento.

Para comenzar, todos los experimentos se llevaron a cabo aplicando el modelo *RoBERTa Large*. Esto nos lleva a tener en cuenta una limitación importante en los sistemas diseñados y desarrollados. Al aplicar un modelo de la arquitectura de *Transformer* se debe tener en cuenta la longitud máxima en número de *tokens* para las entradas del modelo. Con los modelos *Transformer*, hay un límite en la longitud de las secuencias que podemos pasar a los modelos. La mayoría de los modelos manejan secuencias de hasta 512 o 1024 *tokens*, y se bloquean cuando se les pide que procesen secuencias más largas. Por ello, en la fase de aprendizaje, se fijó una longitud máxima de 512 tokens por documento. El documento es la concatenación de los últimos 50 mensajes o publicaciones de un usuario. El ajuste a 50 documentos es debido a un previo análisis donde se sacó en conclusión que había una mayor diferencia lingüística entre los usuarios que presentaban rasgos de ludopatía y los que no, en las últimas publicaciones, y no en las primeras. Por ello se extraen las 50 últimas publicaciones para realizar el entrenamiento de los sistemas, que es el número aproximado de publicaciones que se ajusta mejor al tamaño máximo establecido. El tamaño máximo establecido a 512 es debido a que se desean tener

en cuenta el máximo número de *tokens* posibles, siendo un número mayor de tamaño inviable para los recursos hardware de los que se dispone en este trabajo.

Debido a la limitación del tiempo y la experimentación planteada, los parámetros que se detallan a continuación se fijaron de acuerdo a su valor por defecto o a la mejora de los sistemas de forma general en su búsqueda aplicando la técnica de validación cruzada. La tasa de aprendizaje para el ajuste fino de la red neuronal se fijó en $5 * 10^{-5}$, debido a su valor por defecto para estos modelos. El tamaño del lote se fijó en 8, debido a que un mayor tamaño era inviable para los recursos de los que se disponía y un menor tamaño no daba mejores resultados. Finalmente, el número de épocas de entrenamiento se fijó en una, ya que un número mayor de épocas desencadenaba un sobre-ajuste del sistema a los datos de entrenamiento. El optimizador utilizado fue AdamW, al ser utilizado por defecto por el modelo. Además, para explorar los mejores parámetros y evaluar todo el sistema, se utilizó el método de validación cruzada quíntuple.

Un *Transformer* está formado por varias capas similares, apiladas unas sobre otras. Cada capa tiene una entrada y una salida. En este caso, el modelo pre-entrenado *RoBERTa Large* tiene una última capa de tamaño 1.024 para cada *token*. Por lo que este estado es la salida del modelo y la entrada a la FNN desarrollada para los experimentos.

Experimento 1

Para el experimento 1 se tiene en cuenta como única entrada los 1.024 estados de la salida del modelo. Y se tienen en cuenta dos sistemas distintos, un sistema de regresión (exp.1.1) y un sistema multiclase (exp.1.2).

Experimento 2

Para el experimento 2 se tiene en cuenta como única entrada los 1.024 estados de la salida del modelo más 79 vectores de características agrupadas en cuatro categorías: volumetría, diversidad léxica, complejidad léxica y emociones. Y se tienen en cuenta dos sistemas distintos, un sistema de regresión (exp.2.1) y un sistema multiclase (exp.2.2).

Experimento 3

Para el experimento 3 se tiene en cuenta como única entrada los 1.024 estados de la salida del modelo más 24 vectores de características pertenecientes a la categoría de volumetría. Y se tienen en cuenta dos sistemas distintos, un sistema de regresión (exp.3.1) y un sistema multiclase (exp.3.2).

Experimento 4

Para el experimento 4 se tiene en cuenta como única entrada los 1.024 estados de la salida del modelo más 8 vectores de características pertenecientes a la categoría de diversidad léxica. Y se tienen en cuenta dos sistemas distintos, un sistema de regresión (exp.4.1) y un sistema multiclase (exp.4.2).

Experimento 5

Para el experimento 5 se tiene en cuenta como única entrada los 1.024 estados de la salida del modelo más 13 vectores de características pertenecientes a la categoría de complejidad léxica. Y se tienen en cuenta dos sistemas distintos, un sistema de regresión (exp.5.1) y un sistema multiclase (exp.5.2).

Experimento 6

Para el experimento 6 se tiene en cuenta como única entrada los 1.024 estados de la salida del modelo más 34 vectores de características pertenecientes a la categoría de emociones. Y se tienen en cuenta dos sistemas distintos, un sistema de regresión (exp.6.1) y un sistema multiclase (exp.6.2).

17.1.2. Predicción y evaluación.

Para la predicción y evaluación de los modelos entrenados se han tenido en cuenta los datos de evaluación proporcionados por la organización. De estos datos, solo se han extraído las primeras 50 publicaciones de cada uno de los sujetos para evaluarlos. De esta forma, a cada uno de los sistemas se la pasa únicamente la primera publicación de cada uno de los sujetos existentes y se evalúa dicho sujeto. En una segunda ronda, se pasa la segunda publicación de cada uno de los usuarios a los sistemas y se evalúa al usuario teniendo en cuenta sus dos primeras publicaciones. En una tercera ronda, se tiene en cuenta las tres primeras publicaciones de todos los sujetos existentes para volver a evaluar al usuario. Así, continuarán evaluando a los usuarios los distintos sistemas hasta un total de 50 publicaciones debido a la limitación de tiempo para poder evaluar todos los mensajes de todos los sujetos. En la evaluación se tiene en cuenta que en el momento en el que un sujeto sea evaluado como positivo en presentar rasgos de ludopatía, no se tendrán en cuenta las evaluaciones realizadas posteriormente a este sujeto. Es decir, si el sujeto se continúa evaluado como positivo, o como negativo, en las evaluaciones posteriores a una positiva, no se tendrá en cuenta este resultado.

En la figura 17.1 se muestra un diagrama que muestra una ejemplificación del proceso de predicción.

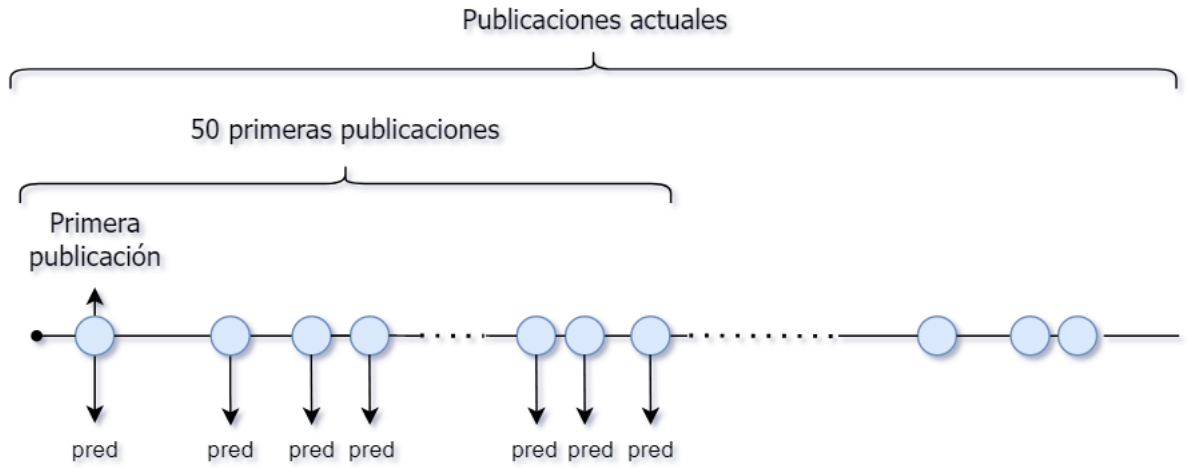

Figura 17.1: Diagrama de representación temporal de publicaciones realizadas por un usuario cualquiera y su predicción por los sistemas.

Por otro lado, dado que la semilla inicial tomada en los sistemas, tanto para el entrenamiento como para la evaluación y predicción, daba grandes diferencias en los resultados obtenidos finalmente, se ha decidido tomar la misma semilla para todos los sistemas, de forma que se puedan evaluar uniformemente.

17.2. Resultados y discusión.

En esta sección se muestran los resultados obtenidos en cada experimento de acuerdo a las métricas definidas anteriormente y se comparan y discuten los resultados para obtener conclusiones.

17.2.1. Evaluación basada en decisiones.

Una de las formas de evaluación de resolución del problema es considerando el problema de decisión binaria, los usuarios tienen que ser etiquetados como positivos o negativos, es decir, según sus publicaciones, deben ser detectados como posibles sufridores del trastorno de ludopatía o no. Además, se tiene en cuenta que cuanto antes detecte el sistema una posible adicción, será mejor, tal y como se refleja con las métricas ERDE5, ERDE50, latencyTP, speed y latency-weightedF1. Además, se evalúan otras métricas como precisión, cobertura o F1.

En la tabla 17.1 se visualizan los resultados de las métricas descritas para k = 1, es decir, para la primera publicación facilitada de cada uno de los usuarios a evaluar, de cada uno de los experimentos llevados a cabo en este trabajo.

Exp.	P	R	F1	ERDE5	ERDE50	latencyTP	Speed	latency-weightedF1
Exp. 1.1	0,657	0,568	0,609	0,018	**0,017**	**1,000**	**1,000**	0,609
Exp. 1.2	0,800	0,494	0,611	0,020	0,013	**1,000**	**1,000**	0,611
Exp. 2.1	0,905	0,469	0,618	0,021	0,021	**1,000**	**1,000**	0,618
Exp. 2.2	0,870	0,494	0,630	0,020	0,020	**1,000**	**1,000**	0,630
Exp. 3.1	0,687	0,568	0,622	0,018	**0,017**	**1,000**	**1,000**	0,622
Exp. 3.2	0,405	0,580	0,477	0,018	0,018	**1,000**	**1,000**	0,477
Exp. 4.1	**0,930**	0,494	0,645	0,020	0,020	**1,000**	**1,000**	0,645
Exp. 4.2	0,717	0,531	0,610	0,019	0,019	**1,000**	**1,000**	0,610
Exp. 5.1	0,843	0,531	**0,652**	0,019	0,018	**1,000**	**1,000**	**0,652**
Exp. 5.2	0,415	**0,605**	0,492	**0,017**	**0,017**	**1,000**	**1,000**	0,492
Exp. 6.1	0,639	0,481	0,549	0,021	0,206	**1,000**	**1,000**	0,549
Exp. 6.2	0,872	0,506	0,641	0,0,20	0,194	**1,000**	**1,000**	0,641

Tabla 17.1: Evaluación basada en decisiones para k = 1.

En la tabla 17.1 se visualizan los resultados de las métricas para k = 50, es decir, para las primeras 50 publicaciones facilitadas de cada uno de los usuarios a evaluar.

Exp.	P	R	F1	ERDE5	ERDE50	latencyTP	Speed	latency-weightedF1
Exp. 1.1	0,496	0,778	0,606	0,015	0,098	**1,000**	**1,000**	0,606
Exp. 1.2	0,803	0,753	0,777	0,015	0,099	**1,000**	**1,000**	0,777
Exp. 2.1	0,813	0,753	0,782	0,015	0,098	**1,000**	**1,000**	0,782
Exp. 2.2	**0,862**	0,691	0,767	0,015	**0,012**	**1,000**	**1,000**	0,767
Exp. 3.1	0,636	0,778	0,700	**0,013**	0,093	**1,000**	**1,000**	0,700
Exp. 3.2	0,418	**0,815**	0,552	**0,013**	0,089	**1,000**	**1,000**	0,552
Exp. 4.1	0,859	0,753	0,803	0,015	0,098	**1,000**	**1,000**	0,803
Exp. 4.2	0,851	0,778	**0,813**	0,014	0,091	**1,000**	**1,000**	**0,813**
Exp. 5.1	0,782	0,753	0,767	0,014	0,099	**1,000**	**1,000**	0,767
Exp. 5.2	0,284	0,790	0,418	0,015	**0,012**	**1,000**	**1,000**	0,418
Exp. 6.1	0,823	0,630	0,713	0,017	0,014	**1,000**	**1,000**	0,713
Exp. 6.2	0,750	0,778	0,764	0,014	0,091	**1,000**	**1,000**	0,764

Tabla 17.2: Evaluación basada en decisiones para k = 50.

En ambas tablas 17.1 y 17.2, se puede ver cómo en aspectos como la velocidad, *speed*, y latencia, *latencyTP*, todos los sistemas que se han desarrollado en este trabajo alcanzan los valores máximos para estas métricas. La velocidad (*speed*), comprendida entre 0 y 1, alcanza su máximo valor en todos los experimentos, lo que significa que los positivos se detectan

velozmente, es decir, la velocidad igual a 1 se aplica para sistemas cuyos verdaderos positivos se detectan justo en la primera publicación. Por otro lado, la latencia (*LatencyTP*), indica el número medio de publicaciones necesarias por el sistema para detectar como positivos los verdaderos positivos. En este caso la media sale 1 en todos los experimentos por lo que, de nuevo, se observa la velocidad de los sistemas para detectar a ciertos sujetos como positivos. Respecto a la métrica ERDE, los resultados parecen ser aceptables ya que el error es muy bajo para todos los sistemas, siendo el valor mínimo alcanzado 0,013 y 0,012 para ERDE5 y ERDE50 respectivamente, y el valor máximo 0,021 y 0,099.

Respecto al número de publicaciones tomadas en cuenta para la predicción, siendo una publicación necesaria en los resultados de la tabla 17.1 y cincuenta publicaciones en los resultados de la tabla 17.2, la precisión (P) parece ser mejor para $k = 1$ mientras que la cobertura (R) mejora notablemente para $k = 50$, es decir, consideramos que teniendo en cuenta un mayor número de publicaciones de cada sujeto, es más fácil determinar si el sujeto padece o no rasgos de ludopatía. Por otro lado, la métrica latency-weightedF1 combina la eficacia y el retraso de la decisión, que, en este caso, para todos los sistemas tiene un valor más o menos similar, siendo, de nuevo, ligeramente mejores los valores obtenidos en la tabla 17.2 considerando un mayor número de publicaciones.

Comparando los sistemas teniendo en cuenta el tipo de estos, es decir, comparando los sistemas de regresión (exp.1.1, exp.2.1, exp.3.1, exp.4.1, exp.5.1 y exp.6.1) con los sistemas de clasificación (exp.1.2, exp.2.2, exp.3.2, exp.4.2, exp.5.2 y exp.6.2) no podemos sacar conclusiones ya que los resultados obtenidos no indican unas diferencias claras entre ellos. Sin embargo, la cobertura suele ser ligeramente mejor en los sistemas de clasificación con respecto a los sistemas de regresión. Esto puede ser debido a que determinar el umbral tras aplicar una función *sigmoide* sobre la salida no es algo que el modelo pueda aprender, ya que está fijado en 0,5, mientras que el uso de la función *softmax* en dos clases permite explorar mejor el espacio de pérdidas.

Con estos resultados, además, no se puede determinar si la integración de características lingüísticas mejora o no las predicciones dadas por este tipo de sistema ya que no se observan grandes diferencias en las tablas. Sin embargo, se puede decir que los resultados son, de forma global, buenos, ya que el principal objetivo de los sistemas aplicados a este tipo de datos consiste en una detección precoz de los sujetos positivos. Es necesario un análisis más detallado para determinar cómo contribuyen las características al rendimiento.

Si comparamos los resultados obtenidos en la tabla 17.2 con los obtenidos por los equipos participantes en la campaña de evaluación contenidos en Martın-Rodilla et al., 2022, teniendo en cuenta que solo se han evaluado a los sujetos con las primeras cincuenta publicaciones de cada uno, se observa que los sistemas desarrollados en este trabajo detectan precozmente a los sujetos positivos, respecto a otros sistemas. Sin embargo, la cobertura tiene aún margen para mejorar con respecto al resto. En el Apéndice A encontramos un artículo relacionado con

la campaña de evaluación donde se incluyeron sistemas similares.

17.2.2. Evaluación basada en la clasificación.

Se recuerda que la otra forma de evaluación de resolución del problema basado en la decisión de clasificación, en lugar de asignar etiquetas de forma binaria, calcula una puntuación de la estimación del riesgo de sufrir el trastorno de ludopatía para cada usuario. Las métricas que se consideran para evaluar esta segunda visión de la tarea son P@10, NDCG@10 y NDCG@100.

En primer lugar, en la tabla 17.3 se visualizan los resultados para k = 1, es decir, para la primera publicación de cada uno de los usuarios.

Exp.	P@10	NDCG@10	NDCG@100
Exp. 1.1	0,900	0,918	0,675
Exp. 1.2	**1,000**	**1,000**	0,673
Exp. 2.1	0,900	0,940	0,656
Exp. 2.2	0,900	0,880	0,630
Exp. 3.1	0,800	0,812	0,656
Exp. 3.2	0,900	0,943	0,681
Exp. 4.1	**1,000**	**1,000**	0,685
Exp. 4.2	0,900	0,932	0,683
Exp. 5.1	**1,000**	**1,000**	0,681
Exp. 5.2	**1,000**	**1,000**	0,651
Exp. 6.1	0,600	0,470	0,523
Exp. 6.2	0,900	0,926	**0,693**

Tabla 17.3: Evaluación basada en la clasificación para k = 1.

En segundo lugar, en la tabla 17.4 se visualizan los resultados para k = 50, es decir, para las primeras 50 publicaciones de cada uno de los usuarios.

Exp.	P@10	NDCG@10	NDCG@100
Exp. 1.1	**1,000**	**1,000**	0,833
Exp. 1.2	0,800	0,752	0,781
Exp. 2.1	**1,000**	**1,000**	0,815
Exp. 2.2	0,900	0,926	0,799
Exp. 3.1	0,800	0,742	0,771
Exp. 3.2	**1,000**	**1,000**	**0,868**
Exp. 4.1	**1,000**	**1,000**	0,844
Exp. 4.2	**1,000**	**1,000**	0,850
Exp. 5.1	0,900	0,880	0,803
Exp. 5.2	**1,000**	**1,000**	0,732
Exp. 6.1	0,800	0,752	0,709
Exp. 6.2	**1,000**	**1,000**	0,840

Tabla 17.4: Evaluación basada en la clasificación para k = 50.

Los resultados obtenidos para la evaluación basada en ranking están contenidos en las tablas 17.3 y 17.4, siendo una publicación tomada en cuenta en los resultados de la tabla 17.3 y cincuenta publicaciones en los resultados de la tabla 17.4. En general se observan unas puntuaciones altas siendo mejores las obtenidas en la segunda tabla.

De nuevo, no podemos sacar conclusiones respecto a los sistemas de clasificación o regresión y respecto a los que integran determinadas características lingüísticas con los que no, pero sí podemos observar cómo tener en cuenta un mayor numero de publicaciones mejora, en general, los resultados en ambas evaluaciones. Además, este tipo de arquitectura sobre la que se apoyan los sistemas desarrollados en este trabajo cumple el requisito de detectar precozmente a los sujetos positivos para este conjunto de datos.

Otro rasgo que se ha identificado en la experimentación es que la semilla inicial que toma el clasificador marca una gran diferencia en el entrenamiento, evaluación y predicción de los modelos. Una semilla puede dar lugar a unos resultados muy buenos mientras que otra puede dar unos resultados con una detección de cero sujetos positivos.

Es necesaria una labor más profunda de investigación, con un mayor tiempo y recursos disponibles para su ejecución, en la que determinar si la integración de características lingüísticas en redes neuronales profundas es un factor determinante en tareas del PLN, como por ejemplo la clasificación de sujetos que se lleva a cabo en este proyecto.

Capítulo 18

Modelo de negocio.

El objetivo de este capítulo consiste en proponer un modelo de negocio para que el proyecto pueda ser comercializado.

El trabajo descrito en el presente trabajo puede entenderse como la solución propuesta a un problema planteado por una organización, el congreso sobre el que se apoya, o como respuesta a un problema identificado en un grupo social, en este caso aquel grupo que presenta el trastorno de ludopatía o podría presentarlo. Sin embargo, el modelo de negocio que se propone para la comercialización futura del producto final obtenido tras la realización del trabajo está enfocado a empresas con intereses similares.

Con los nuevos modelos y sistemas que se han generado en este trabajo se podría desarrollar un sistema que sea capaz de, a través de publicaciones en redes sociales, determinar si un usuario presenta riesgo de padecer ludopatía.

En el ámbito de la IA, concretamente del PLN, nos encontramos con un alto avance en la investigación, sin embargo, hay pocas investigaciones relacionadas con la adicción al juego, por lo que uno de los modelos de negocio posibles para este trabajo es aportar un predictor que nos ayude a comprender los mejores métodos y enfoques que pueden aplicarse, no solo en la detección precoz de signos de ludopatía, sino también en casos similares, como el ciberacoso, la anorexia, la depresión o incluso el suicidio.

Como ejemplo de proyecto investigación en este ámbito concreto, en la Universidad de Jaén[1], el grupo de investigación SINAI[2] está trabajando actualmente en el proyecto Big Hug[3], financiado por la Junta de Andalucía, centrado en la detección precoz de trastornos y comportamientos inadecuados (depresión, ansiedad, trastornos alimentarios, adicción al juego, ideación suicida y ciberacoso) en las redes sociales. También se ha iniciado el proyecto PRECOM, financiado por el Ministerio de Consumo del Gobierno de España, para,

[1] https://www.ujaen.es/
[2] https://sinai.ujaen.es
[3] https://bighug.ujaen.es/

precisamente, la detección precoz de la adicción al juego a partir de la información vertida en redes sociales.

Capítulo 19

Conclusiones y trabajo futuro.

El objetivo de este capítulo consiste en la enumeración de las principales conclusiones obtenidas tras terminar el proyecto y, a demás, proponer mejoras y líneas de trabajo futuras.

19.1. Conclusión.

Al inicio de este proyecto, los sistemas de redes neuronales eran casi desconocidos para mí así como las técnicas de PLN y extracción de características lingüísticas. Por ello, me alegra haber finalizado este proyecto habiendo adquirido un gran volumen de conocimiento sobre este campo, que ahora sé que me apasiona.

Respecto a los objetivos marcados el inicio del proyecto, no ha habido problema a la hora de cumplirlos:

- Gracias a la revisión bibliográfica realizada y al estudio de las distintas tecnologías del lenguaje se ha podido realizar una buena elección de recursos y tecnologías para lo necesario en este proyecto.
- Se llevó a cabo un análisis de las distintas tecnologías capaces de soportar este proyecto y se eligieron las adecuadas permitiéndome adquirir unos mayores conocimientos acerca de estas.
- Se ha desarrollado el necesario análisis, diseño e implementación de los datos y sistemas necesarios en el trabajo.
- Además, la memoria y los distintos manuales referentes a este proyecto se han llevado a cabo en Latex. Esto ha supuesto muchas facilidades.

Al finalizar este trabajo, sacamos como conclusión final, que no se aprecian diferencias entre los sistemas de regresión y clasificación desarrollados. Adicionalmente, tampoco se aprecian

diferencias entre integrar o no características lingüísticas en las redes neuronales profundas utilizadas con los datos usados en este trabajo. Parece que este tipo de modelos utilizados extraen sus propias características y no necesitan de este tipo concreto de integraciones adicionales, teniendo en cuenta este caso. Por lo tanto, una mayor labor de investigación sería necesaria para poder sacar diferencias debido a que numerosos estudios han demostrado una mejora en los sistemas al integrar características lingüísticas, por tanto, los resultados dados en este proyecto pueden significar que es necesario una labor de investigación más profunda donde se tengan en cuenta otros aspectos. Todo ello sería útil en un trabajo futuro.

19.2. Trabajo futuro.

Como todo trabajo no perfecto, este también es mejorable por lo que durante un desarrollo futuro, con más tiempo, podrían añadirse distintas funciones nuevas y mejoras en versiones posteriores. Algunas de las nuevas funcionalidades que se pueden presentar serían:

- Integración de características temporales. Tener en cuenta las características temporales, no solo lingüísticas. Así se puede evaluar otro aspecto muy relevante, a priori, de los datos.
- Añadir más datos estadísticos a los resultados de una prueba. Tener en cuenta otras medidas de evaluación o incluso integrar una evaluación dedicada al impacto ambiental causado en la realización del trabajo.
- Hacer otros pre-procesamientos. Realizar un procesamiento de los datos más exhaustivo, eliminando o reemplazando ciertos términos, por ejemplo.
- Utilizar otro tipo de tecnologías. Podrían llevarse a cabo experimentos haciendo uso de otros modelos o redes como, por ejemplo, *Long Short-Term Memory* (LSTM).
- Utilizar todos los datos proporcionados. En lugar de utilizar solamente los últimos cincuenta *posts* del historial de cada usuario, aplicar otros métodos para hacer uso de todo el conjunto de datos facilitado.

Siglas

AA Aprendizaje Automático.

BERT *Bidirectional Encoder Representations from Transformers*.

CEN Comité Europeo de Normalización.

DCG Ganancia Acumulada Descontada.

DL *Deep Learning*.

DO *Drop Out*.

EDT Estructura de Desglose del Trabajo.

ERDE *Early Risk Detection Error*.

FN *False Negative*.

FNN *Feed-forward Neural Network*.

FP *False Positive*.

GPUs Unidades de Procesamiento Gráfico.

HPC *High Performance Computin*.

IA Inteligencia Artificial.

IDCG Ganancia Acumulada Ideal Descontada.

LSTM *Long Short-Term Memory*.

ML *Machine Learning*.

NDCG Ganancia Acumulada Descontada Normalizada.

PLN Procesamiento del Lenguaje Natural.

ReLU Unidad Lineal Rectificada.

RoBERTa *Robustly Optimized BERT Approach*.

SSH *Secure SHell*.

TN *True Negative*.

TP *True Positive*.

WBS *Work Breakdown Structure*.

Bibliografía

Adoma, A. F., Henry, N.-M., & Chen, W. (2020). Comparative Analyses of Bert, Roberta, Distilbert, and Xlnet for Text-Based Emotion Recognition. *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 117-121. https://doi.org/10.1109/ICCWAMTIP51612.2020.9317379

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bauskar, S., Badole, V., Jain, P., & Chawla, M. (2019). Natural language processing based hybrid model for detecting fake news using content-based features and social features. *International Journal of Information Engineering and Electronic Business*, *11*(4), 1-10.

Bengio, Y. (2009). Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, *2*(1), 1-127.

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. .º'Reilly Media, Inc."

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). Language Models are Few-Shot Learners.

C. I. Rivas, J. F. G., V. P. Corona, & Hernández, L. (2015). Metodologías actuales de desarrollo de software. *Tecnología e Innovación*, *2, No. 5, 980-986*.

Chopra, A., Prashar, A., & Sain, C. (2013). Natural language processing. *International journal of technology enhancements and emerging engineering research*, *1*(4), 131-134.

Choudhary, A., & Arora, A. (2021). Linguistic feature based learning model for fake news detection and classification. *Expert Systems with Applications*, *169*, 114171.

Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. *Proceedings of the 25th international conference on Machine learning*, 160-167.

Conneau, A., & Lample, G. (2019). Cross-lingual language model pretraining. *Advances in neural information processing systems*, *32*.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J. G., Le, Q. V., & Salakhutdinov, R. (2019). Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. *CoRR*, *abs/1901.02860*. http://arxiv.org/abs/1901.02860

Demszky, D., Movshovitz-Attias, D., Ko, J., Cowen, A., Nemade, G., & Ravi, S. (2020). GoEmotions: A Dataset of Fine-Grained Emotions. *58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Devi, T. R., & Reddy, V. S. (2012). Work breakdown structure of the project. *Int J Eng Res Appl*, *2*(2), 683-686.

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR*, *abs/1810.04805*. http://arxiv.org/abs/1810.04805

Dong, G., & Liu, H. (2018). *Feature engineering for machine learning and data analytics*. CRC Press.

García-Díaz, J. A., Jiménez-Zafra, S. M., García-Cumbreras, M. A., & Valencia-García, R. (2022). Evaluating feature combination strategies for hate-speech detection in spanish using linguistic features and transformers. *Complex & Intelligent Systems*, 1-22.

Goutte, C., & Gaussier, E. (2005). A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. *European conference on information retrieval*, 345-359.

Hosting, O. (s.f.). Metodologías del Desarrollo de Software [https://okhosting.com/blog/metodologias-del-desarrollo-de-software/#En_que_consisten_las_Metodologias_de_Desarrollo_de_Software].

Khurana, D., Koli, A., Khatter, K., & Singh, S. (2022). Natural language processing: state of the art, current trends and challenges. *Multimedia Tools and Applications*. https://doi.org/10.1007/s11042-022-13428-4

Krishnamoorthy, S. (2015). Linguistic features for review helpfulness prediction. *Expert Systems with Applications*, *42*(7), 3751-3759.

Kumar, E. (2013). *Natural language processing*. IK International Pvt Ltd.

Liddy, E. D. (2001). Natural language processing. *Encyclopedia of Library and Information Science*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, *abs/1907.11692*. http://arxiv.org/abs/1907.11692

Liu, Z., & Park, S. (2015). What makes a useful online review? Implication for travel product websites. *Tourism management*, *47*, 140-151.

López-Anguita, R., Montejo-Ráez, A., Martínez-Santiago, F. J., & Díaz-Galiano, M. C. (2018). Legibilidad del texto, métricas de complejidad y la importancia de las palabras. *Procesamiento del Lenguaje Natural*, *61*, 101-108.

Losada, D. E., & Crestani, F. (2016). A test collection for research on depression and language use. *International Conference of the Cross-Language Evaluation Forum for European Languages*, 28-39.

Marcos, P. C.

bibinitperiod E. (2001). Procesos Ágiles para el Desarrollo de Aplicaciones Web. *VI Jornadas de Ingeniería del Software y Bases de Datos*.

Marcot, B. G., & Hanea, A. M. (2021). What is an optimal value of k in k-fold cross-validation in discrete Bayesian network analysis? *Computational Statistics*, *36*(3), 2009-2031.

Markovitch, S., & Rosenstein, D. (2002). Feature generation using general constructor functions. *Machine Learning*, *49*(1), 59-98.

Martın-Rodilla, P., Losada, D. E., & Crestani, F. (2022). Overview of eRisk 2022: Early Risk Prediction on the Internet. *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 13th International Conference of the CLEF Association, CLEF 2022, Bologna, Italy, September 5–8, 2022, Proceedings*, *13390*, 233.

McCarthy, P. M., & Jarvis, S. (2010). MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior research methods*, *42*(2), 381-392.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, *26*.

Murarka, A., Radhakrishnan, B., & Ravichandran, S. (2020). Detection and Classification of mental illnesses on social media using RoBERTa. *arXiv preprint arXiv:2011.11226*.

Neupane, P. (2020). Understanding text classification in NLP with Movie Review Example Example [https://www.analyticsvidhya.com/blog/2020/12/understanding-text-classification-in-nlp-with-movie-review-example-example/]. *Data Science Blogathon.*

O. T. Gómez, P. P. R. L., & Bacalla, J. S. (2010). Criterios de selección de metodologías de desarrollo de software. *Facultad de Ingeniería Industrial 13(1): 70-74*.

Pastor, R. T. (2011). Planificación y programación de operaciones. *Perspectivas n.28*.

Pérez, A. (2016). Características y fases del modelo incremental [https://www.obsbusiness.school/blog/caracteristicas-y-fases-del-modelo-incremental].

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training.

Refaeilzadeh, P., Tang, L., & Liu, H. (2009). Cross-validation. *Encyclopedia of database systems*, *5*, 532-538.

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019a). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, *abs/1910.01108*. http://arxiv.org/abs/1910.01108

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019b). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, *abs/1910.01108*. http://arxiv.org/abs/1910.01108

Saravia, E., Liu, H.-C. T., Huang, Y.-H., Wu, J., & Chen, Y.-S. (2018). CARER: Contextualized Affect Representations for Emotion Recognition. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3687-3697. https://doi.org/10.18653/v1/D18-1404

Shinde, P. P., & Shah, S. (2018). A review of machine learning and deep learning applications. *2018 Fourth international conference on computing communication control and automation (ICCUBEA)*, 1-6.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. *CoRR*, *abs/1706.03762*. http://arxiv.org/abs/1706.03762

Zhang, J., Dong, B., & Philip, S. Y. (2020). Fakedetector: Effective fake news detection with deep diffusive neural network. *2020 IEEE 36th international conference on data engineering (ICDE)*, 1826-1829.

Zhang, L., Wang, S., & Liu, B. (2018). Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *8*(4), e1253.

Apéndice A

Publicaciones científicas

SINAI at eRisk@CLEF 2022: Approaching Early Detection of Gambling and Eating Disorders with Natural Language Processing

Alba María Mármol-Romero, Salud María Jiménez-Zafra*, Flor Miriam Plaza-del-Arco, M. Dolores Molina-González, María-Teresa Martín-Valdivia and Arturo Montejo-Ráez

Computer Science Department, SINAI, CEATIC, Universidad de Jaén, 23071, Spain

Abstract

This paper describes the participation of the SINAI team in the eRisk@CLEF lab. Specifically, two of the proposed tasks have been addressed: i) Task 1 on the early detection of signs of pathological gambling, and ii) Task 3 on measuring the severity of the signs of eating disorders. The approach presented in Task 1 is based on the use of sentence embeddings from Transformers with features related to volumetry, lexical diversity, complexity metrics, and emotion related scores, while the approach for Task 3 is based on text similarity estimation using contextualized word embeddings from Transformers. In Task 1, our team has been ranked in second position, with an F1 score of 0.808, out of 41 participant submissions. In Task 3, our team also placed second out of a total of 3 participating teams.

Keywords

Early risk prediction, Gambling detection, Eating disorders detection, Natural Language Processing, Transformers, Volumetry, Lexical diversity, Complexity, Emotion detection, Text similarity

1. Introduction

The large amount of content posted daily on social media has made them a significant source of data for early detection of mental disorders and risky behaviours. The eRisk@CLEF 2022 lab [1] focuses on early risk prediction on the Internet and its goal is to promote the development of automatic systems for the detection of mental disorders such as depression, self-harm or eating disorders. In this edition, three tasks have been proposed:

- Task 1: Early Detection of Signs of Pathological Gambling. It involves sequentially processing writings and detecting as early as possible the first signs of pathological gambling. It is a continuation of the Task 1 proposed for eRisk 2021, but the difference is that in this edition training data has been provided, whereas last year it was an "only test" task. The training data of this edition comprises all test users of the 2021 task.

CLEF 2022: Conference and Labs of the Evaluation Forum, September 5–8, 2022, Bologna, Italy
amarmol@ujaen.es (A. M. Mármol-Romero); sjzafra@ujaen.es (S. M. Jiménez-Zafra); fmplaza@ujaen.es (F. M. Plaza-del-Arco); mdmolina@ujaen.es (M. D. Molina-González); maite@ujaen.es (M. Martín-Valdivia); amontejo@ujaen.es (A. Montejo-Ráez)
0000-0001-7952-4541 (A. M. Mármol-Romero); 0000-0003-3274-8825 (S. M. Jiménez-Zafra); 0000-0002-3020-5512 (F. M. Plaza-del-Arco); 0000-0002-8348-7154 (M. D. Molina-González); 0000-0002-2874-0401 (M. Martín-Valdivia); 0000-0002-8643-2714 (A. Montejo-Ráez)

- Task 2: Early Detection of Depression. It consists of sequentially processing pieces of evidence and detect early traces of depression as soon as possible. It is a continuation of eRisk 2017 pilot task and eRisk 2018 Task 1. The difference between this edition and past editions is in the training set provided. In 2022, the training data is composed of all 2017 pilot task users (training users + test users) plus 2018 Task 1 test users.
- Task 3: Measuring the severity of the signs of Eating Disorders. Its aim is to estimate a user's level of disordered eating from his or her history of posts. For this purpose, for each user, a standard eating disorder questionnaire (EDE-Q) has to be filled in. This task is new in this edition and no training data are provided, that is, it is an "only test" task.

Currently, our research group SINAI[1] is working on the Big Hug project[2] focused on the early detection of disorders and misbehaviors (depression, anxiety, eating disorders, gambling addiction, suicidal ideation and cyberbullying) in online social networks. Therefore, our interest in developing systems as those expected to answer eRisk tasks is high, as it is a perfect playground to test our approaches.

In this sense, our main goals are not only to produce systems reporting high performance but to understand the best methods and approaches that can be applied in similar scenarios. It is not our aim to put as many features and as many systems as possible all together in an ensemble of predictors to gain the top ranking position, but rather to find out the best approaches that can be applied to our project's objectives. The design of online and monitoring tools, as is requested in Task 1, along with the ability to understand user's disorder, the main pursuit in Task 3, fully matches our research interests.

This work presents the participation of our research group, the SINAI team, in *Task 1: Early Detection of Signs of Pathological Gambling* and *Task 3: Measuring the severity of the signs of Eating Disorders*. The rest of the paper is organized as follows. Sections 2 and 3 describe the details of our participation in Task 1 and Task 3, respectively. Each of them is divided into subsections in which, first, we introduce what the task consists of, the data provided and the evaluation measures used. Secondly, the system developed and the methodology used are presented. Thirdly, the experimental setup is detailed. Subsequently, the results obtained and a discussion of them are presented. Finally, Section 4 shows the conclusions obtained after the participation in the eRisk lab and the perspectives for future work.

2. Task 1: Early Detection of Signs of Pathological Gambling

2.1. Task description

This task focuses on early risk detection of gambling addiction by processing posts from social media in strict order of publication. The participant systems had to read the posts (from several users) in the order in which they were created, process them and generate a response in order to get the next posts. The data is composed of 14,627 posts by 81 different subjects categorized as positive in gambling addiction, and about 1M posts by 1,998 subjects not categorized as addicts [1].

[1] https://sinai.ujaen.es
[2] https://bighug.ujaen.es

The task is faced from two different perspectives: as a binary decision problem, and as a ranking (regression) decision problem. As a binary decision problem, posts have to be labelled as positive (label 1, i.e. addiction detected) or negative (label 0, no addiction detected). The earlier the system detects an addiction, the better, as it is reflected with the ERDE and $F_{latency}$ metrics proposed by the organizers and used to evaluate the systems, along with the well known precision, recall and F1 scores. As a ranking decision problem, instead of assigning 0 or 1 labels, a score of the estimation of the risk to suffer such a disorder is computed. Different metrics as the ones used for information retrieval are considered to evaluate this second view of the task (P@10 or NDCG, among others).

2.2. System and methods

In order to address this task, we have followed a supervised learning approach. To train our models, we have used the training dataset provided by the eRisk organizers. This dataset consists of a time series of posts published by different users. Due to the limited sequence length of transformer models used (which will be described later), we made numerous tests to choose the right number of posts that would be more representative of the pathology (from the oldest ones to the newest ones). Finally, the 50 most recent posts were taken after evaluating different sizes.

An important part on which prediction models are based is the extraction of a feature vector from each set of posts per user provided by the organization. The features extracted by our system collect different aspects: volumetry, lexical diversity, complexity metrics, and emotion related scores. We explain in more detail what these features are about and the resources used to produce them.

- Volumetry: we extract the number of words, number of unique words, number of characters, word average length, number of unique lemmas, long average of lemmas and number of each part-of-speech found. This was done with the Python package spaCy [2].
- Lexical diversity: we apply different techniques to measuring lexical diversity, such as simple Type-Token Ratio (TTR), root TTR, log TTR, Maas TTR, Mean Segmental Type–Token Ratio (MSTTR), Moving-Average Type–Token Ratio (MATTR), Hypergeometric Distribution Diversity (HDD) and Measure of Textual Lexical Diversity (MTLD) [3].
- Complexity: we apply different techniques to measuring the text complexity, such as the lexical complexity, Spaulding readability, sentence complexity, automated readability index, height of the dependency tree, punctuation marks, Fernández-Huerta's readability, Flesch-Szigrist readability, Gutierrez's comprehensibility, readability, minimum age of comprehension and SOL metric [4].
- Emotions: this feature is focused on measuring the different emotions that are expressing in posts, such as fear, anger, joy or sadness. We obtain these emotions using two available pre-trained language models based on the Transformer architecture [5]: DistilBERT[3] [6] model fine-tuned on an annotated Twitter corpus on emotions [7] and a BERT[4] [8] model fine-tuned on an annotated Reddit corpus on emotions [9].

[3] https://huggingface.co/bhadresh-savani/distilbert-base-uncased-emotion
[4] https://huggingface.co/bhadresh-savani/bert-base-go-emotion

For our participation in eRisk Task 1 we have trained a total of three models, all supported by the RoBERTa-large [10] linguistic model and feature vectors. Sentence embeddings from RoBERTa are concatenated with the features described above after scaling them into the [0, 1] interval, as can be seen in Figure 1.

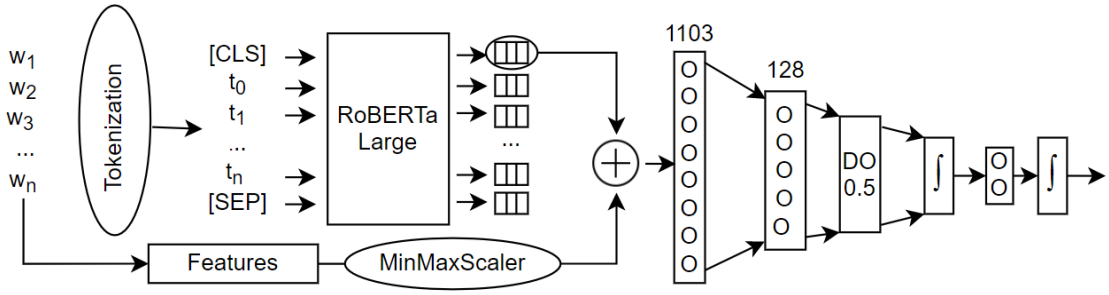

Figure 1: Model architecture for the binary decision task. The input representation is RoBERTa-Large tokenization. Each embedding of the model output is concatenated with its corresponding normalized feature vector. The difference for the classification task is the last FFN with 1 exit instead of 2 exits.

As first step, the process we follow to predict task 1 consists in the extraction of the posts. The last 50 posts of each user are concatenated. This concatenation of posts is the input to the RoBERTa large automodel whose output is concatenated with the previously generated and normalized feature vectors (those about volumetry, complexity, lexical diversity and emotions). Once all features have been concatenated, a Feed Forward Neural Network (FFNN) with one hidden layer is applied to generate the final predictions. The first layer of the final FFN has an input dimension of 1103 (1024 from the embeddings vector generated by RoBERTa and 79 additional features). The output of the layer has a size of 128. These outputs are passed through a dropout layer (with probability 0.5) during learning and, finally, a ReLU activation function is also applied before feeding a last FFN with 128 inputs and 1 or 2 outputs (the final decision classes depending on the configuration of the run, as described below).

The system was implemented using the Python packages scikit-learn[11], Transformers[5] and PyTorch[12], and trained on a 2xGPU NVIDIA V100 server.

2.3. Experimental setup

In the learning phase, a maximum length of 512 tokens per document was set. The document is the concatenation of the last 50 posts of a user (as described before). The learning rate for fine-tuning the neural network was set to 5e-5, the batch size was 8 and the number of train epochs was one. The optimizer used was AdamW. To explore the best parameters and evaluate the whole system, 5-fold cross validation approach was followed.

All runs follow the same structure. The runs differ in minor configuration aspects, but that triggered major changes in the prediction.

- **Run 0**. This run consists of running a regression model in which the set of posts used has no pre-processing. The result of this model marks the score that a user has to be

considered a potential gambler or not. Users with a score lower than 0.5 will not be considered potential gamblers, while those with a score equal to or higher than 0.5 will be considered potential gamblers.

- **Run 1**. In this run, the only difference with respect to run 0 is the processing of the concatenated posts. Before extracting the feature vector, the URLs found in the texts are removed (also brackets and parentheses containing URLs).
- **Run 2**. This run is very similar to run 0. The set of texts used does not have any processing, however, the model built is a binary classification model (instead of a regression one), so the result of this model indicates the score that a user has to be considered and not considered a pathological gambler. The model generates a score for the two considered classes (addicted or not), which will define the final class after a softmax transformation.

2.4. Results and discussion

Salient results have been achieved with the approaches explored by our team. From the reported results provided by the organizers, we have extracted our scores, which are shown in Tables 1 and 2.

Table 1
Results of SINAI team for Task 1 in decision-based evaluation

Run	P	R	$F1$	$ERDE_5$	$ERDE_{50}$	$latency_{tp}$	$speed$	$latency_w F1$
0	0.425	0.765	0.546	0.015	0.011	1.0	1.000	0.546
1	0.575	0.802	0.670	0.015	0.009	1.0	1.000	0.670
2	0.908	0.728	0.808	0.016	0.011	1.0	1.000	0.808

In decision-based evaluation, the value of F1 score obtained by our Run 2 (0.808) is the second highest among all submissions by participants (41 submissions were reported in total). Our ERDE values are the first and second highest (0.015 and 0.016). In terms of $speed$ and $latency_{TP}$ we also reach top values. The system that generated Run 2 is highly performant in this task, despite the limited amount of resources involved.

Table 2
Results of SINAI team for Task 1 in ranking-based evaluation (only 1 writing results reported)

Run	P@10	NDCG@10	NDCG@100
0	0.10	0.19	0.56
1	0.70	0.65	0.62
2	1.00	1.00	0.70

Regarding ranking-based evaluation (see Table 2), our team reaches a third position with values of 1.0 for both, P@10 and NDCG@10, and 0.7 for NDCG@100. It can be noticed that these scores are always zero for our system above 1 writing. We believe this is due to the limited number of submissions that our system needs to trigger an alert. Therefore, not enough scores are provided for a number of submissions above 100 for the ranking evaluation to be feasible.

Our results indicate that the binary configuration is more convenient to train the system. The reason may be that, in the fist two configurations, determining the threshold after applying

a sigmoid function on the output logit is not something the model can learn, as it is fixed on 0.5, while using softmax on two classes allows for a better exploration through the loss space. Additionally, further analysis is needed to determine how non-embeddings features contribute to the performance beyond pure end-to-end models.

3. Task 3: Measuring the severity of the signs of Eating Disorders

3.1. Task description

In previous years, early detection of anorexia signs has been conducted in eRisk@CLEF [13, 14, 15]. This year, however, Task 3 "Measure the severity of signs of eating disorders" is proposed for the first time, which consists of an estimation of the level of characteristics associated with an eating disorder diagnosis from a user's writing history. For each user, a submission history is provided and participants are required to automatically fill in a standard eating disorder questionnaire. An important aspect is that no training data is provided to address this task.

The questionnaires are defined on the basis of the Eating Disorder Examination Questionnaire (EDE-Q). This questionnaire is designed to assess the range and severity of multiple characteristics associated with eating disorders. From it, the organizers have used only the questions 1-12 and 19-28. It employs four subscales: restraint (RS), preoccupation with eating (ECS), preoccupation with shape (CSC), preoccupation with weight (WCS), and a global score (GED). To obtain a particular subscale score, the ratings for the relevant questions (numbered in evaluation metrics) are added together and the sum divided by the total number of questions forming the subscale. To obtain an overall or "global" score, the four subscales scores are summed and the resulting total divided by the number of subscales (i.e. four).

The official evaluation metrics of the competition are the following: MZOE, MAE, MAE_{macro}, GED, RS, ECS, SCS, WCS. Refer to the overview of the task [1] for a detailed explanation of these measures.

3.2. System and methods

In this section, the methodology developed to address Task 3 is described.

First of all, we performed a preprocessing step to select those posts corresponding to the last 28 days of the user's history to follow the instructions given in the questionnaire: "The following questions are concerned with the past four weeks (28 days) only".

Once these posts were selected, the next step was to define the methodology used to answer the questionnaire. For this aim, we followed a method based on text similarity estimation using contextualized word embeddings from Transformers. Specifically, we computed the similarity between each post in a user's history with each question in the EDE-Q Questionnaire. This value ranges from 0 to 1 where a value close to 0 indicates that the post and the question are not similar while a value close to 1 indicates the opposite.

After computing this similarity, we adopted a heuristic to answer the questions given. For this aim, we first differentiated two types of questions that can be found in the eating disorder questionnaire: day-based questions and scale-based questions. The day-based questions are

those whose answers are of the form: no days, 1-5 days, 6-12 days, etc., while the answers to the scale-based questions are as follows: not at all, slightly, moderately, etc.

On the one hand, to answer the day-based questions, we calculated the number of days that the user talks about the topic of the question. For this, the posts whose similarity with the question is greater than a threshold value (0.4 for run 1, 0.35 for run 2 and 0.375 for run 3) are selected. On the selected posts, the date of the first and the last post is chosen and the difference in days between the two is calculated. After that, we selected the option that matched the number of days obtained:

0. NO DAYS (0 days)
1. 1-5 DAYS (1<= days <= 5)
2. 6-12 DAYS (6 <= days <= 12)
3. 13-15 DAYS (13 <= days <= 15)
4. 16-22 DAYS (16 <= days <= 22)
5. 23-27 DAYS (23 <= days <= 27)
6. EVERY DAY (days >= 28)

On the other hand, in order to answer the scale-based questions, for each question, we first selected the user's post with the highest computed similarity value. Then, we defined the following intervals to select the scale:

0. not at all (0 to 0.1)
1. slightly (0.1 to 0.2)
2. slightly (0.2 to 0.3)
3. moderately (0.3 to 0.4)
4. moderately (0.4 to 0.5)
5. markedly (0.5 to 0.6)
6. markedly (0.6 to 1)

This heuristic distinguishes between low and high similarity values so that the higher the similarity, the more likely the response is associated with having the disorder. For instance, if the similarity score computed is 0.65, we chose the answer "markedly" for the associated question.

3.3. Experimental setup

As part of our participation in Task 3, three runs have been submitted according to the system developed. They differ in the similarity value between the post and the day-based questions. For **run 1**, we selected those posts whose similarity is higher than 0.4. For **run 2**, we established this value at 0.35. Finally, for **run 3** we increased slightly the value to 0.375. We aimed to observe the difference in the response selected to the questionnaire given by the organizers according to the similarity computed.

Both the experiments in the pre-evaluation and evaluation phases were run on a compute node equipped with a single Tesla-V100 GPU with 32 GB of memory. We used the spaCy

sentence transformers (spacy-sentence-bert) library[5] to make use of the transformer model RoBERTa with the default parameters.

3.4. Results and discussion

The results obtained with the approaches explored by our team are shown in Table 3. The 3 runs have provided similar results, which was to be expected since the only difference between them is in the similarity value established between the posts and the day-based questions. However, the approach that provided the best results was run 2 with the lowest similarity value considered, 0.35. This indicates that perhaps we should relax this value when identifying the set of posts that are related to each question.

Table 3
Results of SINAI team for Task 3 in ranking-base evaluation

Run	$MZOE$	MAE	MAE_{macro}	GED	RS	ECS	SCS	WCS
1	0.85	2.65	2.29	2.63	3.29	2.35	2.98	2.40
2	0.87	2.60	2.23	2.42	3.01	2.21	2.85	2.31
3	0.86	2.62	2.22	2.54	3.15	2.32	2.93	2.36

In Table 4, we show the best run of each participating team and the 3 evaluations given by the organizers assigning all answers to 0 ("all 0"), 6 ("all 6") and "average" which is obtained from the average of the participants' submissions.

Table 4
Results of best run for team for Task 3 in ranking-base evaluation

Run	$MZOE$	MAE	MAE_{macro}	GED	RS	ECS	SCS	WCS
IISERB_2	0.92	2.18	1.76	1.74	2.00	1.73	2.03	1.92
SINAI_2	0.87	2.60	2.23	2.42	3.01	2.21	2.85	2.31
RELAI_3	0.83	3.15	2.70	3.26	3.04	2.72	4.04	3.61
all 0	0.81	3.36	2.96	3.68	3.69	3.18	4.28	3.82
all 6	0.67	2.64	3.04	3.25	3.52	3.72	2.81	3.28
average	0.88	2.72	2.22	2.69	2.76	2.20	3.35	2.85

The MZOE measure reflects whether the answers given by the system to complete the questionnaire were correct or not. The closer its value is to 1, the higher the fraction of incorrect predictions. Overall, we can see that the systems presented by the 3 teams make more than 80% of incorrect predictions, which reflects the difficulty of the task. In our case, the 3 runs provide a similar score, but run 1, based on the similarity heuristic with the highest threshold, 0.4, achieves the higher number of correct answers. However, the MAE and MAE_{macro} measures (range from 0 to infinity, the lower the better) indicate that the responses given by our system are not very far from the real responses, specially those from the run 2. The RS, ECS, SCS, and WCS scores allow us to identify the questions in which our system failed the most, being the most difficult to predict those related to food restriction (questions 1, 2, 3, 4, and 5) and those concerning shape (questions 6, 8, 23, 10, 26, 27, 28, and 11). If we compare our best run with the

[5] https://spacy.io/universe/project/spacy-sentence-bert

average results provided by the organisers, we can see that we have a better record except for the RS score, which is above average, and the ECS score which is similar to the average.

In past editions (2018 and 2019) [13, 14], tasks related to anorexia detection have been presented where the challenge consists of sequentially processing pieces of evidence and detecting early traces of anorexia as soon as possible. However, this year, although the disorder to be focused is the same (anorexia), the formulation of the task is different, being the first time that the challenge aimed at developing an automatic system to fill a standard eating disorder questionnaire based on the evidence found in the user's history. It is worth noting that although the evaluation measures proposed this year are different, this task presents a greater challenge compared to the past editions where the maximum value achieved in terms of F1 score was .71 [14]. In addition, this is also reflected in the low number of teams that this task has attracted this year (3 compared to 13 teams in 2019).

4. Conclusions and future work

This paper describes our participation as SINAI team in Task 1 and Task 3 of the eRisk@CLEF 2022 edition. The former is the continuation of the first edition in 2021 and aims to detect signs of pathological gambling as soon as possible, while the latter is a new task that focused on measuring the severity of eating disorders signs. For Task 1, we have developed regression and classification models using state-of-the-art pre-trained language models based on Transformers. Besides, for the classification model, we explored a variety of linguistic features including volumetry, lexical diversity, complexity, and emotion detection, achieving the second position among the participants with this model. For Task 3, as no training data was provided by the organizers, we decided to rely on text similarity estimation using word embeddings from Transformers and designing our heuristics. The results achieved in this task as well as the low participation show the difficulty of addressing this type of problem with an automatic system. This fact demonstrates the need to continue investing efforts in this important task.

In future work, we plan to analyze in depth the linguistic features considered in Task 1 to understand to what extent they contribute to the detection of signs of pathological gambling along with further data pre-processing. For Task 3, we would like to perform an error analysis to identify the main weaknesses of our system, as well as explore other Natural Language Processing models.

Acknowledgments

This work has been partially supported by Big Hug project (P20_00956, PAIDI 2020) and WeLee project (1380939, FEDER Andalucía 2014-2020) funded by the Andalusian Regional Government, and LIVING-LANG project (RTI2018-094653-B-C21) funded by MCIN/AEI/10.13039/501100011033 and by ERDF A way of making Europe. Salud María Jiménez-Zafra has been partially supported by a grant from Fondo Social Europeo and Administración de la Junta de Andalucía (DOC_01073). Flor Miriam Plaza-del-Arco has been partially supported by a grant from the Ministry of Science, Innovation and Universities of the Spanish Government (FPI-PRE2019-089310).

References

[1] J. Parapar, P. Martín-Rodilla, D. E. Losada, F. Crestani, Overview of eRisk 2022: Early Risk Prediction on the Internet, in: Experimental IR Meets Multilinguality, Multimodality, and Interaction. 13th International Conference of the CLEF Association, CLEF 2022. Springer International Publishing, Bologna, Italy., 2022.

[2] M. Honnibal, I. Montani, spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing, 2017. Unpublished software application. https://spacy. io.

[3] P. M. McCarthy, S. Jarvis, MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment, Behavior Research Methods 42 (2010) 381–392. URL: https://doi.org/10.3758/BRM.42.2.381. doi:10.3758/BRM.42.2.381.

[4] R. López-Anguita, A. Montejo-Ráez, F. J. Martínez-Santiago, M. C. Díaz-Galiano, Legibilidad del texto, métricas de complejidad y la importancia de las palabras, Procesamiento del Lenguaje Natural 61 (2018) 101–108. URL: http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/view/2018-61-11. doi:10.26342/2018-61-11, number: 0.

[5] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, A. M. Rush, Transformers: State-of-the-art natural language processing, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics, Online, 2020, pp. 38–45. URL: https://aclanthology.org/2020.emnlp-demos.6. doi:10.18653/v1/2020.emnlp-demos.6.

[6] V. Sanh, L. Debut, J. Chaumond, T. Wolf, Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, ArXiv abs/1910.01108 (2019).

[7] E. Saravia, H.-C. T. Liu, Y.-H. Huang, J. Wu, Y.-S. Chen, CARER: Contextualized affect representations for emotion recognition, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 3687–3697. URL: https://www.aclweb.org/anthology/D18-1404. doi:10.18653/v1/D18-1404.

[8] J. D. M.-W. C. Kenton, L. K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of NAACL-HLT, 2019, pp. 4171–4186.

[9] D. Demszky, D. Movshovitz-Attias, J. Ko, A. Cowen, G. Nemade, S. Ravi, GoEmotions: A dataset of fine-grained emotions, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, 2020, pp. 4040–4054. URL: https://aclanthology.org/2020.acl-main.372. doi:10.18653/v1/2020.acl-main.372.

[10] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized BERT pretraining approach, CoRR abs/1907.11692 (2019). URL: http://arxiv.org/abs/1907.11692.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, Journal of machine learning research 12 (2011) 2825–2830.

[12] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin,

N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[13] D. E. Losada, F. Crestani, J. Parapar, Overview of erisk 2018: Early risk prediction on the internet (extended lab overview), in: Proceedings of the 9th International Conference of the CLEF Association, CLEF, 2018, pp. 1–20.

[14] D. E. L. F. C. J. Parapar, Overview of erisk at CLEF 2019: Early risk prediction on the internet (extended overview), in: Conference and Labs of the Evaluation Forum, 2019.

[15] F. M. P. del Arco, P. López-Úbeda, M. C. Díaz-Galiano, L. A. U. López, M. T. M. Valdivia, Integrating umls for early detection of sings of anorexia, in: CLEF, 2019.

Apéndice B

Diccionario de la EDT

Nivel	Código	Nombre	Descripción
1	0	TFM	Trabajo de Fin de Máster
2	1	Identificación y definición	Investigación inicial
2	2	Gestión	Documentación y planificación
2	3	Desarrollo	Construcción de sistemas
2	4	Ejecución	Experimentación sobre el problema
3	1.1	Problema	Definición del problema
3	1.2	Alcance	Definición del alcance
3	1.3	Metodología	Definición de la metodología
3	1.4	Fuentes de información	Investigación necesaria en el proyecto
3	1.5	Recursos	Identificación de recursos necesarios
3	1.6	Tecnologías	Identificación de tecnologías necesarias
3	2.1	Documentación	Documentación de todo el proyecto
3	2.2	Planificación	Planificación de todo el proyecto
3	3.1	Análisis de datos	Análisis de los datos de partida
3	3.2	Ingeniería de caracts.	Extracción de caracts. lingüísticas de los datos
3	3.3	Impl. de tecnologías lingüísticas	Desarrollo de software y sistemas
3	4.1	Procesamiento de datos	Procesamiento de los datos de partida
3	4.2	Entrenamiento	Entrenamiento de los experimentos
3	4.3	Evaluación	Evaluación de los experimentos
3	4.4	Análisis resultados	Análisis de los resultados finales
4	1.1.1	Objetivos	Definición de objetivos a alcanzar
4	1.1.2	Hipótesis	Definición de hipótesis
4	1.1.3	Restricciones	Identificación de restricciones
4	2.2.1	Económica	Planificación económica del proyecto
4	2.2.2	Temporal	Planificación temporal del proyecto

Tabla B.1: Diccionario de la Estructura de Desglose del Trabajo.