



Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

*Sistema de detección inteligente de usuarios de Twitter
interesados en la temática de defensa*

Grado en Ingeniería Mecánica

ALUMNO: María Fernández Torrejón

DIRECTORES: Norberto Fernández García
Milagros Fernández Gavilanes

CURSO ACADÉMICO: 2022-2023

Universida_{de}Vigo



Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

*Sistema de detección inteligente de usuarios de Twitter
interesados en la temática de defensa*

Grado en Ingeniería Mecánica
Intensificación en Tecnología Naval
Cuerpo General

UniversidadeVigo

RESUMEN

El continuo y sostenido crecimiento de las redes sociales es un fenómeno que ha adquirido una importancia transcendental en la sociedad contemporánea. La presencia masiva de las redes sociales en la vida diaria de millones de personas a nivel global ha transformado los modos de interacción social, las dinámicas de comunicación, el consumo de información y de noticias, etc.

Tanto es así, que la red social Twitter se ha situado como una de las mayores plataformas de difusión de noticias en tiempo real existentes en el mundo. Esto la convierte en una herramienta de gran valor para la vigilancia y obtención de inteligencia en el ámbito de la defensa.

Todas estas circunstancias han propiciado la necesidad de obtener un sistema de detección inteligente de usuarios de Twitter de interés para el ámbito de la defensa. Se trata de una solución basada en técnicas de procesamiento de lenguaje natural, minería de datos y aprendizaje automático con el objetivo principal de desarrollar un clasificador capaz de discernir si un usuario publica o no contenidos de interés en el ámbito en cuestión. Se procede, a su vez, a realizar una comparativa entre la herramienta implementada y uno de los modelos de lenguaje de gran tamaño (LLM) más afamados y relativamente recientes: el *chatbot* denominado ChatGPT.

PALABRAS CLAVE

Twitter, Defensa, Clasificador, Inteligencia artificial, *Machine Learning*.

AGRADECIMIENTOS

En primer lugar, me gustaría mostrar mi más sincero agradecimiento a mis tutores, D. Norberto Fernández García y Dña. Milagros Fernández Gavilanes, por el esfuerzo, tiempo invertido, orientación y constante apoyo a lo largo de todo el desarrollo del presente proyecto.

A toda la promoción 423-153, en agradecimiento a estos cinco años en la Escuela Naval Militar en los cuales, día a día, se han convertido en mi segunda familia. En especial a mi amigo Ángel, por ser un ejemplo de perseverancia y tenacidad desde el primer día.

A Juanje, por su infinita resiliencia y por enseñarme lo verdaderamente importante.

A mi familia. A mis padres y a mis hermanas, Marta y Pilarín, por su paciencia y apoyo incondicional y fundamental durante todos estos años. Especialmente agradecer a mis hermanas por ser el mejor ejemplo que podría tener.

¡Muchas gracias!

CONTENIDO

Contenido	1
Índice de Figuras	3
Índice de Tablas.....	6
1 Introducción y objetivos.....	7
1.1 Contextualización y motivación.....	7
1.2 Objetivos y metodología	9
1.3 Estructura de la memoria	9
2 Estado del arte	11
2.1 OSINT	11
2.1.1 Historia de OSINT	11
2.1.2 Ventajas y desventajas del empleo de OSINT.....	12
2.1.4 Fuentes OSINT	13
2.1.5 Herramientas OSINT	13
2.2 Twitter.....	14
2.2.1 Historia de Twitter	14
2.2.2 Terminología de Twitter	16
2.2.3 API de Twitter	18
2.3 Empleo de herramientas OSINT en Twitter	19
2.3.1 Herramientas OSINT para Twitter	21
2.4 Detección de comunidades en Twitter	23
2.5 Inteligencia artificial	25
2.5.1 Machine Learning.....	26
2.5.2 Aplicaciones del Machine Learning	27
2.5.3 Métodos de Machine Learning	28
2.5.4 Aprendizaje automático supervisado	28
2.5.5 Modelos de lenguaje	32
2.5.5.1 OpenAI y ChatGPT-3.5	33
2.6 Trabajos académicos relacionados	35
2.6.1 Herramientas de OSINT para la obtención de información en Twitter sobre el conflicto Ucrania-Rusia.....	35
2.6.2 Detección de comunidades en redes: Algoritmos y aplicaciones	35
3 Desarrollo del TFG.....	37
3.1 Instalación de la máquina virtual	38

3.2 Instalación de <i>Python</i> y <i>Sublime Text</i>	39
3.3 Corpus de Wikipedia.....	40
3.4 Generación de corpus para entrenamiento del clasificador	42
3.4.1 API de Wikipedia	42
3.4.2 Selección de categorías.....	43
3.4.3 Obtención de artículos para la generación del corpus	45
3.5 Obtención de tuits de un usuario.....	52
3.5.1 API de Twitter	52
3.5.2 Librería Tweepy.....	54
3.5.3 Obtención de los 100 últimos tuits de un usuario y preprocesador de texto	55
3.6 Implementación del clasificador y entrenamiento	57
3.6.1 Justificación de la elección del algoritmo.....	57
3.6.2 Corpus para el entrenamiento. Obtención de embeddings	58
3.6.3 Entrenamiento del clasificador	60
4 Prueba y validación	64
4.1 Matriz de confusión y métricas empleadas para evaluar el clasificador.....	64
4.2 Evaluación inicial del clasificador con artículos de Wikipedia	66
4.3 Evaluación del clasificador con tuits de usuarios	67
4.4 Evaluación de respuestas obtenidas en ChatGPT	72
5 Conclusiones y líneas futuras	76
5.1 Conclusiones	76
5.2 Líneas futuras	77
6 Bibliografía.....	78
Anexo I: Implicaciones sociales, económicas y ambientales	85
Anexo II: Reflexiones éticas y sociales.....	90

ÍNDICE DE FIGURAS

Figura 1-1 Histórico reciente de usuarios activos diarios monetizables (mDAU). [9]	8
Figura 1-2 Ejemplos de tuits sobre la guerra ruso-ucraniana. (Autoría propia)	8
Figura 2-1 Fases del proceso de investigación OSINT. (Autoría propia)	13
Figura 2-2 OSINT Framework. [26]	14
Figura 2-3 Primer tuit de la historia. [29]	15
Figura 2-4 Cuenta de Twitter del Ministerio de Defensa. (Autoría propia)	17
Figura 2-5 Tuit del Ministerio de Defensa. (Autoría propia)	17
Figura 2-6 Estructura de información en Twitter. (Autoría propia)	18
Figura 2-7 Herramienta "Temas" en Twitter. (Autoría propia)	19
Figura 2-8 Búsqueda avanzada en Twitter. (Autoría propia)	20
Figura 2-9 Grafo de SNA. [38]	21
Figura 2-10 TWINT. [41]	21
Figura 2-11 Truth Nest. [44]	22
Figura 2-12 Twitter Audit. [51]	23
Figura 2-13 Descripción gráfica del concepto de Girvan y Newman. [53]	24
Figura 2-14 Dendrograma. (Autoría propia)	24
Figura 2-15 Gráfico de comunidades solapadas. [53]	25
Figura 2-16 <i>Machine Learning</i> . [60]	27
Figura 2-17 Metodología de aprendizaje automático supervisado. (Autoría propia)	29
Figura 2-18 Ejemplo del algoritmo de árbol de decisión. [70]	30
Figura 2-19 Representación gráfica del algoritmo <i>Random Forest</i> para clasificación. [72]	30
Figura 2-20 Ejemplo del cálculo del hiperplano y el margen en SVM. [74]	31
Figura 2-21 Algoritmo KNN con K=4. (Autoría propia)	32
Figura 2-22 Interfaz de ChatGPT. [76]	34
Figura 2-23 Meses transcurridos hasta llegar a 100 millones de usuarios. [80]	35
Figura 3-1 Metodología para la implementación del sistema de detección. (Autoría propia)	37
Figura 3-2 Nueva máquina virtual. (Autoría propia)	38
Figura 3-3 Configuración máquina virtual. (Autoría propia)	38
Figura 3-4 Índice TIOBE para lenguajes de programación. [86]	39
Figura 3-5 Comando para instalar <i>Python</i> . (Autoría propia)	40
Figura 3-6 Comando para instalar <i>Sublime Text</i> . (Autoría propia)	40
Figura 3-7 Comando para instalar <i>Apache Beam</i> y <i>mwparsersfromhell</i> . (Autoría propia)	40
Figura 3-8 Comando para instalar la biblioteca <i>datasets</i> . (Autoría propia)	40

Figura 3-9 Código de <i>Python</i> para la creación del índice <i>idx_by_title</i> . (Autoría propia)	41
Figura 3-10 Código de <i>Python</i> para obtener un artículo de Wikipedia mediante el título. (Autoría propia)	41
Figura 3-11 Formato JSON de los artículos en el corpus de Wikipedia. [88].....	41
Figura 3-12 Código de <i>Python</i> para guardar el diccionario en un archivo. (Autoría propia).....	42
Figura 3-13 Código de <i>Python</i> para abrir el diccionario. (Autoría propia)	42
Figura 3-14 Estructura base para la obtención de artículos. (Autoría propia).....	43
Figura 3-15 Categoría de Wikipedia " <i>National security</i> " (Seguridad nacional). [95]	44
Figura 3-16 Proceso de obtención de artículos a partir de una categoría. (Autoría propia).....	45
Figura 3-17 Código de <i>Python</i> para obtener artículos y subcategorías pertenecientes a una categoría. (Autoría propia).....	46
Figura 3-18 Archivo de texto <i>titulos_si</i> . (Autoría propia).....	48
Figura 3-19 Archivo de texto <i>subcat_si</i> . (Autoría propia)	48
Figura 3-20 Proceso de obtención de artículos a partir de subcategorías (II). (Autoría propia)	48
Figura 3-21 Código de <i>Python</i> para automatizar el procesamiento de subcategorías. (Autoría propia)	49
Figura 3-22 Código de <i>Python</i> para eliminar títulos repetidos. (Autoría propia)	50
Figura 3-23 Proceso de obtención de los artículos en base a los títulos seleccionados. (Autoría propia)	50
Figura 3-24 Código de <i>Python</i> para volcar el corpus en el fichero <i>articulos_json</i> . (Autoría propia)	51
Figura 3-25 Fichero <i>articulos_json</i> . (Autoría propia)	51
Figura 3-26 Crear una cuenta de usuario en Twitter. (Autoría propia)	52
Figura 3-27 Twitter API. (Autoría propia)	53
Figura 3-28 Pasos para crear un proyecto en el portal de desarrolladores. (Autoría propia)	53
Figura 3-29 Claves de autenticación para desarrolladores. (Autoría propia).....	54
Figura 3-30 Tweepy. [97].....	55
Figura 3-31 Comando para instalar la librería de <i>Python Tweepy</i> . (Autoría propia)	55
Figura 3-32 Proceso para la obtención y limpieza de los últimos cien tuits de un usuario. (Autoría propia)	55
Figura 3-33 Código de <i>Python</i> para la obtención de los cien últimos tuits de un usuario y preprocesado de texto. (Autoría propia)	56
Figura 3-34 Comando para instalar la librería <i>emoji</i> . (Autoría propia).....	57
Figura 3-35 Comando para instalar la librería <i>LangDetect</i> . (Autoría propia).....	57
Figura 3-36 Comando para instalar la librería <i>Googletrans</i> . (Autoría propia).....	57
Figura 3-37 Proceso de obtención de <i>embeddings</i> de artículos de Wikipedia. (Autoría propia)	59
Figura 3-38 Código de <i>Python</i> para obtener <i>embeddings</i> de los artículos de Wikipedia y su almacenamiento en un archivo CSV. (Autoría propia).....	59

Figura 3-39 Proceso de obtención y entrenamiento del clasificador. (Autoría propia).....	60
Figura 3-40 Código de <i>Python</i> para entrenar el clasificador con el conjunto de datos de Wikipedia. (Autoría propia).....	61
Figura 3-41 Proceso general del sistema de detección de usuarios pertenecientes al ámbito de defensa. (Autoría propia)	62
Figura 4-1 Matriz de confusión con posibles salidas del clasificador. (Autoría propia).....	64
Figura 4-2 Código de <i>Python</i> para obtener los parámetros de las predicciones del clasificador ya entrenado con el conjunto de datos de Wikipedia. (Autoría propia)	66
Figura 4-3 Fichero <i>tuits_buenos</i> . (Autoría propia).....	68
Figura 4-4 Proceso de obtención de <i>embeddings</i> de los tuits publicados por usuarios seleccionados. (Autoría propia).....	69
Figura 4-5 Código de <i>Python</i> para la obtención de los <i>embeddings</i> de los tuits. (Autoría propia)..	69
Figura 4-6 Obtención de <i>embeddings</i> . (Autoría propia).....	70
Figura 4-7 Código de <i>Python</i> para evaluar el clasificador entrenado mediante el fichero <i>embeddings_tuits.csv</i> . (Autoría propia)	71
Figura 4-8 Proceso de clasificación mediante ChatGPT-3.5. (Autoría propia)	73
Figura 4-9 Código de <i>Python</i> para clasificar a los usuarios mediante ChatGPT. (Autoría propia) .	74
Figura A1-1 Coste derivado del empleo de la API de OpenAI. (Autoría propia).....	88

ÍNDICE DE TABLAS

Tabla 3-1 Ranking TIOBE para lenguajes de programación a lo largo de los años. [86].....	39
Tabla 3-2 Categorías principales establecidas por el IPTC. (Autoría propia).....	44
Tabla 3-3 <i>Namespace</i> (ns) de las páginas de Wikipedia. (Autoría propia).....	47
Tabla 3-4 Limitaciones de los niveles de acceso para desarrolladores. (Autoría propia)	54
Tabla 3-5 Comandos para instalar las librerías desde el terminal. (Autoría propia).....	58
Tabla 4-1 Probabilidad de pertenecer a cada clase para cada instancia. (Autoría propia)	66
Tabla 4-2 Resultados de la evaluación del clasificador con el corpus de Wikipedia. (Autoría propia).....	67
Tabla 4-3 Usuarios de Twitter seleccionados para evaluar el clasificador. (Autoría propia)	68
Tabla 4-4 Estructura de las variables contenidas en los diccionarios empleados en el clasificador. (Autoría propia).....	70
Tabla 4-5 Resultados obtenidos de la evaluación del clasificador mediante tuits de 40 usuarios. (Autoría propia).....	72
Tabla 4-6 Clasificación realizada por ChatGPT-3.5. (Autoría propia)	75
Tabla A1-1 Horas totales invertidas en el desarrollo del proyecto. (Autoría propia)	86
Tabla A1-2 Horas semanales invertidas en el desarrollo del proyecto I. (Autoría propia)	86
Tabla A1-3 Horas semanales invertidas en el desarrollo del proyecto II. (Autoría propia).....	87
Tabla A1-4 Horas semanales invertidas en el desarrollo del proyecto III. (Autoría propia)	88
Tabla A1-5 Desglose del coste total de mano de obra. (Autoría propia)	88

1 INTRODUCCIÓN Y OBJETIVOS

1.1 Contextualización y motivación

El Allied Joint Intelligence de la OTAN, en 2001 [1], definía en la AJP-2.0.2001 las técnicas OSINT (*Open Source Intelligence*, Inteligencia de fuentes abiertas) de la siguiente manera: “*La inteligencia derivada de una amplia gama de recursos abiertos, como la radio, la televisión, los periódicos, los libros; a los que el público tiene acceso*”.

De esta y muchas otras definiciones, se extrae que OSINT se refiere, principalmente, a las herramientas que facilitan la búsqueda e investigación sobre cierto tema concreto mediante el empleo de fuentes de acceso público o abiertas, con el fin de analizar y correlacionar lo obtenido convirtiéndolo así en información útil [1]. Cabe destacar, que no se trata pues, de técnicas de inteligencia que empleen procedimientos de perturbación, infiltración o interceptación [2].

Este tipo de técnicas encuentran su uso en ámbitos tales como el financiero, el tecnológico o en defensa, entre otros. Este último será el propósito para el que se emplearán a lo largo del proyecto. Si bien, encuentra su máximo desarrollo en aspectos relacionados con la ciberseguridad: prevención de ciberataques, reconocimiento de un *pentesting* (test de intrusión), o en test de ingeniería social recurriendo a redes sociales u otros documentos donde poder encontrar toda la información [3] relacionada con un usuario en concreto. [4]

Twitter es una plataforma social de *microblogging* (microblogueo) que sintetiza ventajas de diferentes plataformas tales como: mensajería instantánea, diferentes redes sociales, blogs, etc. Consiste en un servicio de comunicación bidireccional en el cual, cada usuario tiene la capacidad de publicar información diversa de manera sencilla y sobre todo rápida, lo que la convierten en una de las redes sociales más importantes de la actualidad, situándose en decimoquinta posición en cuanto a la popularidad de redes sociales a nivel mundial [5]. En España en el año 2022, un 85% de las personas con edades entre los 12-70 años hacían uso de redes sociales; de los cuales un 42% empleaban Twitter. Esta red social se sitúa en quinta posición en el ranking nacional de empleo de redes sociales, por detrás de Instagram y YouTube, ambas con un 66% [6].

Dicha plataforma, se caracteriza por el empleo de los conocidos como tuits (*tweets*): mensajes con una extensión límite de 280 caracteres (cabe destacar que usuarios “*blue*”, o *premium*, no tienen esta restricción) en los cuales cada usuario tiene la posibilidad de adjuntar enlaces, vídeos, imágenes o encuestas, por ejemplo. Este formato destaca por su fácil desarrollo y lectura, obteniendo, de esta forma, una diversa naturaleza de los mensajes: desde aquellos personales a publicidad, información corporativa, noticias de carácter internacional, eventos, etc. Cabe destacar que el número medio de tuits publicados cada día en todo el mundo es de 500 millones. [5]

Esta plataforma de comunicación bidireccional fue comprada por el multimillonario Elon Musk el 27 de octubre del pasado año 2022 por una cifra de 44.000 millones de dólares. [7] En cuanto a número de usuarios se refiere, a pesar de que esta plataforma social sea la que posea un crecimiento más lento, finalizó el año 2022 con un total de 1.3 billones de cuentas. En julio del pasado año, Twitter publicó un documento destinado a inversores en el cual afirmaba poseer un total de 237.8 millones de usuarios activos diarios monetizables (mDAU), lo que supone un incremento de más del 20% respecto al pasado año 2021 [8], como se puede apreciar en la siguiente imagen (Figura 1-1).

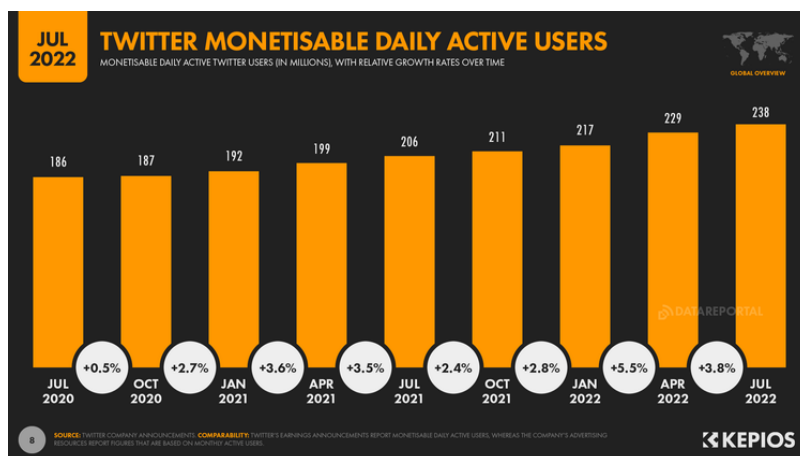


Figura 1-1 Histórico reciente de usuarios activos diarios monetizables (mDAU). [9]

En el podio de países que más emplea dicha red social encontramos a Estados Unidos, seguido de Japón y de la India; situándose España en la decimotercera posición con un total de 4,1 millones de usuarios. [7] Cabe destacar los datos demográficos relativos a esta plataforma, siendo, tan solo un 27,3% de mujeres y un 72,7% de hombres en cuanto a usuarios globales se refiere. [9]

Debido, entre otros, a su alcance global (como se ha mencionado), así como a la inmediatez con la que cualquier noticia es capaz de difundirse en apenas segundos, Twitter se ha convertido en uno de los medios de difusión de información, eventos, noticias, ideologías, etc. más utilizados en los últimos años. Basta con remitirse al reciente conflicto que estalló el pasado 24 de febrero de 2022 entre Ucrania y Rusia, el cual ha ido retransmitiéndose día a día mediante esta red social (véase Figura 1-2). Han sido muchas las cuentas creadas a raíz del estallido de dicha guerra con el fin de mantener informada a la población, así como con objetivos estratégicos de los diferentes bandos.

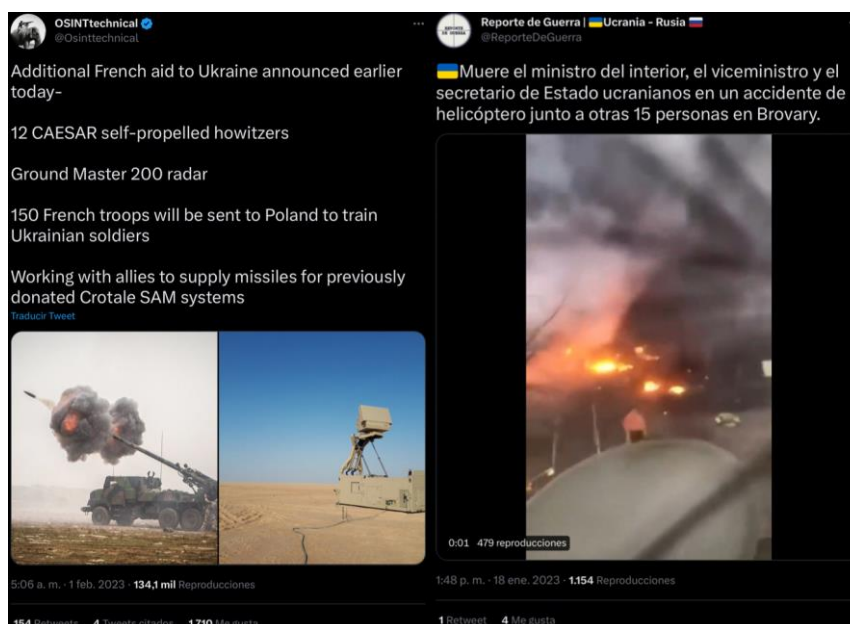


Figura 1-2 Ejemplos de tuits sobre la guerra ruso-ucraniana. (Autoría propia)

Esta plataforma encuentra su máximo potencial de empleo como fuente abierta de información. La llegada de una nueva era, tecnológicamente hablando, ha traído consigo consecuencias como la sustitución de fuentes abiertas tradicionales (periódicos, revistas, radio, folletos publicitarios, etc.), por aquellas de acceso mediante Internet, de las que forman parte las redes sociales y, en concreto, Twitter.

El masivo crecimiento de esta aplicación conlleva la existencia de millones de usuarios, no obstante, no todos ellos serán de utilidad para realizar una investigación acerca de un campo en concreto. Es por ello que nace la necesidad de desarrollar una herramienta que permita la filtración de perfiles relevantes para la temática seleccionada; en este caso, defensa.

Dadas las dimensiones y el alcance de esta red social son muchos los que han mostrado interés en la obtención de información a través de ella. Con este fin se han comenzado a diseñar diferentes herramientas para analizar las publicaciones y datos de cierta relevancia que en ella se exponen. Para la obtención de esta información de relevancia es necesario el empleo de tecnologías, algoritmos y metodologías de inteligencia artificial, pues se trata de operar con ingentes cantidades de datos que no podrían ser procesados manualmente.

1.2 Objetivos y metodología

El principal objetivo de este proyecto final de grado radica en el desarrollo de una herramienta software que permita detectar perfiles de usuarios de la red social Twitter que publican contenidos de interés en el ámbito de la defensa, lo que permitiría a posteriori seguir a estos usuarios con la finalidad de recibir información potencialmente relevante en dicho ámbito.

Para la consecución de este objetivo se emplearán diferentes técnicas y métodos de procesamiento de lenguaje natural y aprendizaje automático (también conocido como *Machine Learning*). En particular, se propone entrenar un clasificador que permita evaluar los contenidos (tuits o, en inglés, *tweets*) que publican los usuarios para discernir entre una clase u otra; es decir, si publica o no contenidos de interés para el ámbito de defensa.

El clasificador se entrenará haciendo uso de un corpus previamente construido tomando como base artículos de la enciclopedia en línea Wikipedia, seleccionados a través de un exhaustivo proceso. A continuación, se evaluará el funcionamiento de dicho clasificador utilizando conjuntos de tuits extraídos de varios perfiles de usuario de la red social Twitter.

La validación de los resultados obtenidos tras las diferentes pruebas en el clasificador es imprescindible para cerrar el ciclo del desarrollo de la herramienta. Se llevará a cabo, a su vez, una comparación con los resultados alcanzados por modelos de inteligencia artificial de lenguaje de gran tamaño, en concreto, el *chatbot* de ChatGPT.

1.3 Estructura de la memoria

Una vez presentados, en líneas generales, los objetivos a alcanzar durante el desarrollo del presente proyecto, así como expuesta la antesala y el contexto en el que este trabajo se enmarca, se procede a realizar un breve repaso sobre la estructura que se va a seguir en la redacción de la presente memoria.

Esta memoria se divide en cinco capítulos principales, una bibliografía en la cual se recogen las fuentes consultadas, y una serie de anexos con documentación referente a las repercusiones medioambientales, éticas y económicas que el desarrollo de esta herramienta puede traer consigo.

- Capítulo 1: En este se contextualiza, en primer lugar, el entorno de trabajo para el desarrollo del proyecto. Se enumeran los objetivos a alcanzar y se estructura (en este preciso apartado) la memoria.
- Capítulo 2: Se procede a exponer de manera detallada los antecedentes del proyecto. Se comienza hablando sobre OSINT profundizando en su historia y las ventajas y desventajas que

este tipo de inteligencia de fuentes abiertas presenta. A continuación, se introduce la red social Twitter y las técnicas de inteligencia de fuentes abiertas que se emplean para analizar dicha red social. Se contextualiza también el termino inteligencia artificial, así como el aprendizaje automático y los algoritmos que de él derivan. Finalmente se hace una breve reseña sobre trabajos científicos previos que guarden algún tipo de relación con el presentado en esta memoria.

- Capítulo 3: Explicación detallada de cada uno de los pasos necesarios a seguir para poder lograr la consecución del clasificador que permitirá la catalogación de cada uno de los usuarios, así como el código necesario para la obtención de los tuits que se precisarán para dicha clasificación.
- Capítulo 4: Se evalúa y valida el clasificador analizando los resultados obtenidos mediante una serie de métricas que en este mismo epígrafe se explicarán. La primera evaluación se realiza con el mismo corpus con el que se procede en el anterior epígrafe al entrenamiento del clasificador, siendo la segunda evaluación la definitiva tomando como entrada tuits obtenidos de distintas cuentas de Twitter.
- Capítulo 5: En el último punto, se hace una síntesis de los objetivos, el desarrollo y los resultados obtenidos señalando cuáles de los propuestos en el primer epígrafe han sido conseguidos y, en caso de que alguno de ellos no haya sido posible lograrlo indicar el por qué. Se presentarán también posibles líneas futuras de investigación como continuación o mejora del actual trabajo.

Por último, se recogen las fuentes utilizadas, así como los anexos mencionados.

2 ESTADO DEL ARTE

2.1 OSINT

2.1.1 Historia de OSINT

A pesar del hecho de que el término OSINT fue acuñado, y tal y como actualmente es conocido, desarrollado en el año 1942; se podría considerar que tuvo sus inicios, en cierto modo, mucho antes. Nos remontamos a la antigua Roma donde, por aquel entonces, ya existían documentos de acceso público, así como foros y pregones donde todo el mundo podía enterarse de cualquier tipo de mensaje que se deseara promulgar. Los gobernantes consultaban esas fuentes abiertas en busca de información que les proporcionase cualquier tipo de inteligencia o ventaja contra sus potenciales enemigos. [1]

Actualmente, el formato de las fuentes abiertas ha cambiado, sin embargo, la esencia sigue siendo la misma. El término OSINT tiene sus orígenes en la Segunda Guerra Mundial, concretamente en la creación del departamento *Research and Analytics* perteneciente al OSS (*Office of Strategic Services*, Oficina de Servicios Estratégicos). Su principal finalidad era recopilar cualquier tipo de información de acceso público mediante la escucha de emisiones de radios extranjeras o la recopilación de periódicos del Eje; considerando esta investigación imprescindible y una de las mayores fuentes de información para la decisiva victoria de los Aliados. Estos últimos, conscientes de que podría producirse en dirección inversa, comenzaron a utilizar la prensa como fuente de desinformación contra sus enemigos. Cabe destacar, asimismo, que el empleo de estas herramientas como fuente de inteligencia, supuso un punto de inflexión durante el transcurso de la Guerra Fría. Es por ello, que se concluye que los orígenes de las técnicas conocidas como OSINT, fueron eminentemente militares. [1]

Con el paso del tiempo, estas técnicas fueron ganando importancia y los diferentes estados fueron reparando en las ventajas que podían obtener en los diferentes conflictos que se sucedieron: la invasión de la Bahía de Cochinos, la Guerra de Vietnam, la Guerra del Golfo, etc. Sin embargo, la gran revolución que se fue fraguando durante los años 90 con respecto al empleo de dichas herramientas, no se produjo hasta el siglo actual cuando la popularización de las computadoras se hizo flagrante y comenzaron a tener un alcance prácticamente global. [1]

A su vez, se generalizó también la conexión a Internet, lo que implicó que la mayor parte de las personas tenían acceso a información y noticias a nivel internacional. Uno de los mayores beneficios que la globalización de Internet trajo consigo, fue el hecho de convertir a cada uno de los usuarios en una fuente de información, así como facilitar el intercambio de esta, dando lugar a un afluyente sin fin de datos. Esto conllevó que la fase crítica de la obtención de inteligencia fiable fuese el procesamiento de información y no tan solo su mera obtención.

El empleo de OSINT se ha generalizado. Hoy en día, cualquier persona que posea unos conocimientos mínimos, y tenga interés en un tema en concreto, puede averiguar, con el simple acceso a fuentes abiertas en Internet, múltiple información.

2.1.2 Ventajas y desventajas del empleo de OSINT

El importante crecimiento de este tipo de herramientas que permiten obtener información de fuentes abiertas evidencia la existencia de ciertas ventajas que han fomentado dicho desarrollo. Las principales son: [10]

- La posibilidad de emplear estas técnicas para cualquier tipo de investigación requerida, independientemente del objetivo de esta.
- Este tipo de herramientas implican menos riesgo, así como una notable comodidad, pues se puede realizar la recopilación de datos desde una oficina sin necesidad de desplazarse.
- A su vez, la posibilidad de obtención de información de forma gratuita y la facilidad de acceso, incrementa el número de usuarios que cada vez optan más por este tipo de fuentes.
- La multitud de fuentes supone una actualización constante y aporta veracidad, pues brinda la posibilidad de comparar. [1]

Todo ello tiene sus desventajas, pues la existencia de incontables fuentes de información implica, en gran parte, poca claridad. Uno de los mayores problemas a la hora de realizar una investigación recurriendo a herramientas OSINT es el filtrado de datos y la jerarquización de estos. Entre la abrumadora cantidad de fuentes abiertas, existen muchas poco fiables, con información errónea. La clave radica en saber discernir entre información certera y la que no lo es. [10]

2.1.3 Fases del proceso de investigación OSINT

A pesar de que prácticamente todo el mundo sepa, presuntamente, indagar en la red y utilizar herramientas OSINT para la obtención de información, existen pocas personas que lo sepan llevar a cabo de manera profesional. Cabe destacar la existencia de un método compuesto por una serie de etapas [4] [11] para lograr la realización de una investigación exhaustiva. Dichas etapas se describen brevemente a continuación (Figura 2-1):

1. Establecer objetivos y fijar requisitos: Consiste en definir las condiciones y requerimientos a cumplir durante el transcurso de la búsqueda, así como los fines últimos que esta persigue.
2. Identificar las fuentes más relevantes: Llevar a cabo un análisis de todas las fuentes abiertas existentes que aporten información de interés y corroborar la fiabilidad de estas.
3. Obtención de la información a estudiar: Concentrar los datos adquiridos de las fuentes seleccionadas y extraer aquella que sea de mayor relevancia para la investigación.
4. Procesamiento y análisis: Tratar la información previamente seleccionada y generar inteligencia mediante su estudio. Permite la jerarquización de los datos en función del nivel de importancia.
5. Presentación de la investigación realizada: Proporcionar formato a los datos y conclusiones obtenidas para que puedan ser comprensibles y útiles, con el fin de ser llevadas a la práctica. Esta última fase es determinante a la hora de realizar una indagación.

La herramienta a desarrollar en el presente trabajo de fin de grado se relaciona con la segunda etapa: la búsqueda de fuentes, las cuales serán en este proyecto los diferentes usuarios de Twitter, de los que obtener potencial información de interés; tarea que, a día de hoy, resulta muy compleja debido principalmente a la inmensa cantidad de usuarios que publican datos en dicha plataforma.

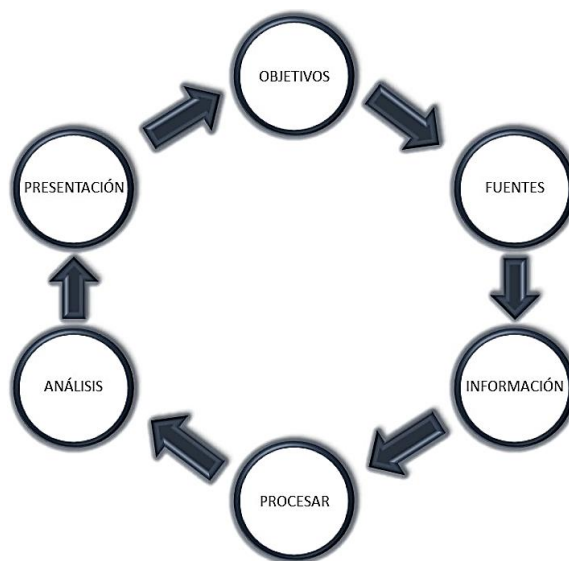


Figura 2-1 Fases del proceso de investigación OSINT. (Autoría propia)

2.1.4 Fuentes OSINT

Se denomina fuente OSINT a cualquier origen del cual se pueda extraer información relevante sobre nuestra investigación u objetivo, por ejemplo: [11] [10]

- Motores de búsqueda: Dentro de estos se encuentran buscadores como Google, Bing, etc. El correcto empleo y explotación de estos, permitirá al usuario realizar una filtración de la información clara y concisa. Esta técnica es conocida como Google Hacking o Google Dorking.
- Redes sociales: El uso de este tipo de aplicaciones a nivel global conlleva, en su mayoría, la publicación de muchos aspectos de la vida de las personas, información relativa a sus gustos, viajes pasatiempos, etc. Sin embargo, también es una fuente de información sobre noticias acaecidas en cualquier parte del mundo que se difunden en tan solo unos segundos. Otra de las consecuencias acerca del empleo de redes sociales, es la publicación de imágenes o vídeos que pueden revelar de manera casi inapreciable información relevante.
- Uso de un sistema operativo destinado a OSINT: Existen distribuciones creadas para favorecer el desarrollo de estas herramientas tales como Osintux [12] y Huron. Existen, a su vez, aquellas destinadas a la ciberseguridad en general como Kali Linux [13] y Parrot Os Security [14].
- Contenido tradicional: Fuentes comúnmente utilizadas a lo largo de los años tales como: periódicos, televisión, anuncios, infografías, etc.

2.1.5 Herramientas OSINT

Con el objetivo de hacer un efectivo empleo de este tipo de fuentes abiertas, se han desarrollado múltiples de herramientas entre las que cabe destacar, como más empleadas, las siguientes [11]:

- Sherlock [15]: Relacionada con las redes sociales, nos permite a través de un nombre de usuario encontrar todas las plataformas sociales en las que posee una cuenta.
- Phoenifoga [16]: A partir de la búsqueda de un número telefónico nos reporta el país, compañía, la seguridad que tiene, etc.
- Thatsthem [17]: Empleo del correo electrónico personal para encontrar información sobre dicha persona.
- Melissa Data [18]: Aplicación que permite la obtención de datos de la propiedad de una vivienda a través de la propia dirección del inmueble.

- DNS Lookup [19]: Partiendo de un nombre de dominio, obtener todos los registros DNS asociados.
- IpVoid [20]: Adquisición de información a partir de una dirección IP proporcionada.
- DNSQuery [21]: En base al nombre de un dominio, se pueden conocer los sitios web que compartan *hosting* (alojamiento web) con el sitio web al que pertenece dicho dominio.
- FOCA: Esta herramienta basa su funcionamiento en buscadores como Google, Bing y DuckDuckGo [22]. Su principal finalidad es la extracción de archivos ofimáticos asociados a un dominio en concreto, los que posteriormente examinará mediante un análisis de metadatos.
- Maltego [23]: Recopila información acerca de un objetivo/ usuario concreto y la muestra en un grafo para su posterior análisis.
- Shodan [24]: Comúnmente conocido como “El Internet de los Hackers”. Facilita al usuario hallar equipos conectados a Internet mediante la aplicación de diferentes filtros. De este modo, es posible localizar rúters, dispositivos IoT, webcams, diferentes servidores, etc.
- Creepy [25]: Consiste en una herramienta desarrollada con propósitos de geolocalización cuya principal función es la recopilación de diferentes ubicaciones extraídas de cuentas en sitios web o redes sociales tales como: Twitter, Facebook, etc. Permite, a su vez, la exportación de un mapa con la representación de dichos lugares.

Cabe destacar que todas ellas se encuentran almacenadas en un repositorio denominado OSINT Framework (Ver Figura 2-2):

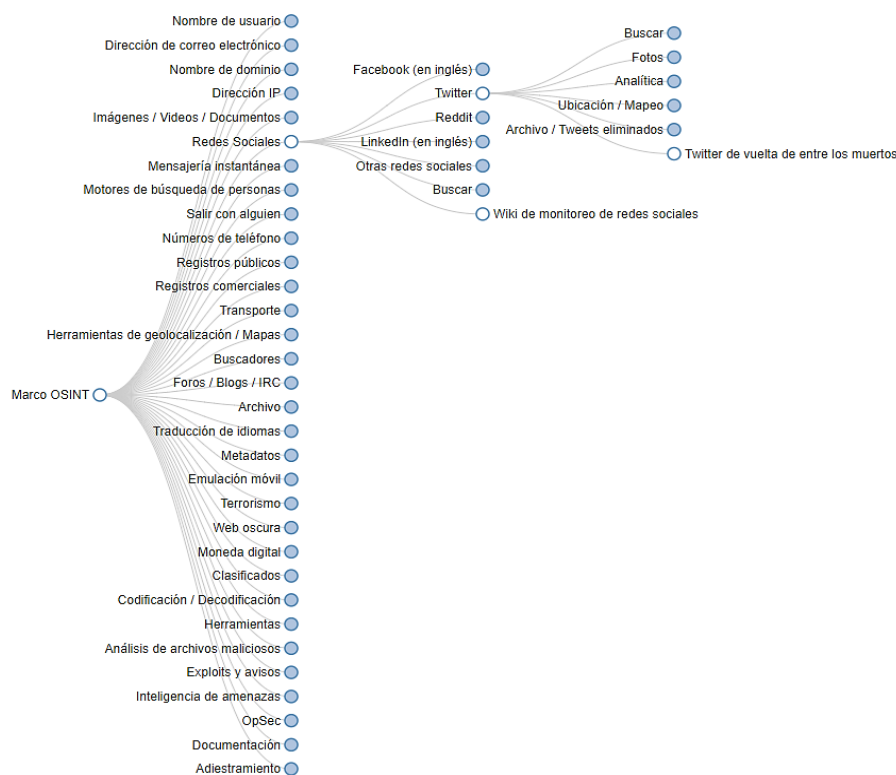


Figura 2-2 OSINT Framework. [26]

2.2 Twitter

2.2.1 Historia de Twitter

Noah Glass, un programador de origen estadounidense desarrolló en 2004 un proyecto denominado Odeo cuya principal idea era dejar un mensaje en formato MP3 en la Nube. Odeo, fue convirtiéndose progresivamente en una plataforma de *podcast* (anglicismo generalizado en el uso y sin

un equivalente univocal en español. Puede adaptarse fácilmente a nuestra lengua mediante la aplicación de la tilde: pódcast [27]) a la que se uniría Evan Williams. Coetáneamente, Apple lanzó al mercado la aplicación iTunes. Es por ello, que, Williams como CEO de Odeo concluyó que la *startup* (empresa emergente) necesitaba reinventarse en pos de eludir el hundimiento. Emerge entonces la idea de emplear algo similar a los conocidos como SMS para crear conversaciones entre reducidas comunidades de personas. [28]

En la concepción de este proyecto se encuentra el nacimiento de la red social conocida como Twitter. Sus inicios se remontan al año 2006, cuando después de barajar diferentes nombres (Twiiit, Twitch, Stat.us, etc.), Glass decidió que se denominaría 'Twtr', procurando imitar el trino de un pájaro. La falta de vocales es debida a una costumbre de suprimirlas en los nombres de las diferentes aplicaciones que, por aquel entonces y hoy en día se desarrollan (Tumblr, Flickr, etc.). El conocido como primer tuit fue redactado por Jack Dorsey el 21 de marzo de ese mismo año y decía "*Just setting up my twtr*" (Configurando mi twtr) (Figura 2-3). [28]



Figura 2-3 Primer tuit de la historia. [29]

Una de las principales características de este servicio fue la restricción de caracteres a 140, pues al ser una plataforma basada en mensajería móvil, este era el límite estándar que los operadores establecían por aquel entonces. [28] [30]

Tras unos meses críticos en los que muchos de los inversores originales fueron perdiendo el interés al ver el proyecto original fracasar, Obvious Corporation fue fundado en octubre del mismo año por Jack Dorsey, Evan Williams y Biz Stones, quienes compraron los derechos de Twitter a Odeo. Fue Williams quien se puso al frente del nuevo proyecto. [28] [31]

Twitter no se emancipa como empresa independiente hasta 2007. Comienza a crecer el número de usuarios de manera exponencial, así como el número de tuits publicados, llegando a alcanzar 70.000 diarios. Es entonces cuando gana un prestigioso premio: South by Southwest. Este mismo año, un usuario emplea una almohadilla (#) con el fin de identificar un tema en uno de sus tuits, inventando inconscientemente la etiqueta (*hashtag*), la cual tardaría varios meses en implementarse de manera oficial en la red social. [28]

Dorsey fue expulsado por decisión unánime de la empresa. El número de usuarios fue incrementando de manera exponencial, aumentó un 1.300% en el año 2009 llegando a ser un total de 71 millones de usuarios [28]. El número fue incontrolable y comenzó una etapa de declive en la empresa, pues se producían colapsos y caídas de manera periódica. Williams y Stone no supieron lidiar con dicho problema. Además, a pesar de contar con tal cantidad de usuarios, no se generaban apenas beneficios.

Como consecuencia, y ante la incapacidad de hacer frente a la situación, la junta de Twitter optó por el despido de Williams y el regreso de Jack Dorsey, quien ocuparía el puesto de presidente ejecutivo. Biz Stone renunció a su puesto y se desentendió de la empresa en 2011. Fue en 2015 cuando Dorsey fue nombrado CEO de la empresa, puesto al que renunció en noviembre de 2021. No obstante,

retiene actualmente su propiedad del 2.4% de la red social [28], aún después de la compra por Elon Musk en octubre de 2022. [32]

Los errores fueron subsanados y el crecimiento de Twitter prosiguió. Fueron apareciendo los denominados *trending topic* (tendencias), cuyo objetivo era remarcar los temas más populares en cada momento para que los usuarios puedan acceder a todos los tuits que tengan relación. Aparecieron oficialmente, las etiquetas precedidas por una almohadilla (#) permitiendo, de este modo, el rastreo de dichas tendencias. En 2010 surgió una de las actualizaciones que en apenas minutos se viralizó, la posibilidad de redactar un tuit que contenga archivos tales como imágenes, vídeos o contenidos pertenecientes a otras redes sociales. Finalmente, en 2017, la restricción de caracteres aumentó a 280 y se introdujeron distintas funcionalidades como la posibilidad de responder a otros tuits, la mención de diferentes usuarios, los retuits, la posibilidad de crear encuestas, etc. [28] (Más información a este respecto se incluye en la sección 2.2.2)

Twitter se convirtió en uno de los canales más importantes para la transmisión y seguimiento de diferentes noticias de carácter global, así como un medio de comunicación más para la política y la cultura, entre otros, principalmente debido a la rapidez con la que se conseguía llegar a millones de personas. Cabe destacar que hitos tales como las agitadas elecciones de 2009 en Irán fueron altamente beneficiosos para la empresa. De su importancia en tales eventos políticos habla el hecho de que, en junio de 2019, la red social desveló la existencia de cuentas falsas vinculadas con gobiernos de Irán y Rusia. [31]

A comienzos de abril de 2022, Elon Musk, posicionado como el hombre más rico del mundo compró el 9,2% de las acciones de la empresa, pasando a ser el accionista mayoritario. Este aseguró que lo hacía convencido del potencial de la plataforma, con el objetivo de poner fin a las incesantes censuras que se estaban produciendo, para convertirla en un sitio que fomentase la libertad de expresión. Tras muchos conflictos derivados de la afirmación que Musk realizó acerca de la compra de la empresa, y las diferentes negociaciones que tuvieron lugar en consecuencia, el 27 de octubre de 2022 el multimillonario firmó la compra por 44.000 millones de dólares.

El nuevo propietario ha llevado a cabo múltiples cambios como el despido de prácticamente todo el equipo directivo y asegura que quiere convertir la red social en la más respetable del mundo. A día de hoy, esta red social es considerada una de las plataformas digitales con más actividad del mundo. Twitter se ha convertido en una herramienta esencial en la vida de muchas personas, empresas e incluso gobiernos. Muchos aseguran que la cualidad inherente que ha impulsado su crecimiento es su simpleza.

2.2.2 Terminología de Twitter

Para el correcto desarrollo y comprensión del proyecto caben definir los siguientes términos [5]:

- Tuitero: usuario registrado en la plataforma. Son identificados por los conocidos como nombres de usuarios los cuales van precedidos por una arroba (@nombredeusuario). (Véase Figura 2-4)
- Tuit (*Tweet*): cada uno de los mensajes publicados por un tuitero. Tiene una restricción vigente de un máximo de 280 caracteres (no incluido material multimedia). (Véase Figura 2-5)
- Retuit (*Retweet*): Consiste en la republicación de un tuit que ha sido previamente publicado por otro usuario. (Véase Figura 2-5)
- Seguidor (*Follower*): Es el usuario que sigue una cuenta y que, por tanto, puede ver todos los tuits que publica. (Véase Figura 2-4)
- Siguiendo (*Following*): La acción de seguir a otro usuario. (Véase Figura 2-4)
- Etiqueta (*Hashtag*): Permite facilitar la búsqueda en un ámbito concreto con ayuda un término precedido por una almohadilla (#). (Véase Figura 2-5)
- Tendencia (*Trending Topic*): Es la palabra, etiqueta o conjunto de palabras más mencionadas en la red social en un determinado período de tiempo.

- **Mención:** Cualquier tuit que contenga el nombre de usuario de otra persona como parte del cuerpo del mensaje. (Véase Figura 2-5)
- **Línea del tiempo (Timeline):** Lugar en el cual la aplicación muestra a cada usuario, por orden cronológico, los tuits publicados por los diferentes usuarios a los que sigue.
- **Me gusta (Like):** Función que permite a los diferentes usuarios mostrar que le ha gustado el tuit publicado por un tuitero. Queda representado por un corazón. (Véase Figura 2-5)



Figura 2-4 Cuenta de Twitter del Ministerio de Defensa. (Autoría propia)

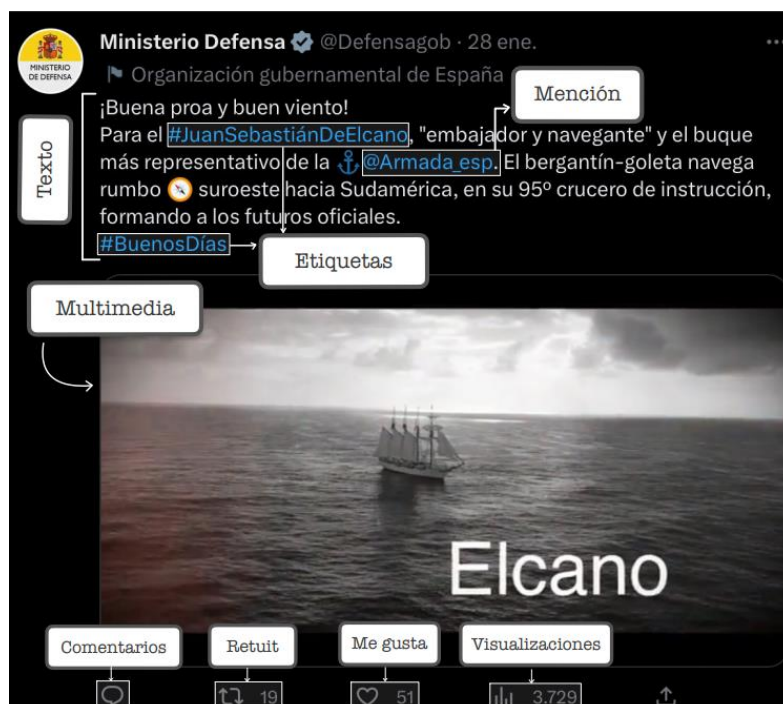


Figura 2-5 Tuit del Ministerio de Defensa. (Autoría propia)

Para llevar a cabo una correcta y exhaustiva investigación es necesario comprender el diagrama de relaciones entre los diferentes términos y funcionalidades que la red social ofrece (véase Figura 2-6). De este modo es posible tener un punto de partida desde el cual comenzar la exploración con herramientas OSINT. Son de crucial importancia las relaciones seguidor-seguido, así como las interacciones que entre estos desarrollen (retuits, contestaciones, menciones, me gusta, etc.).

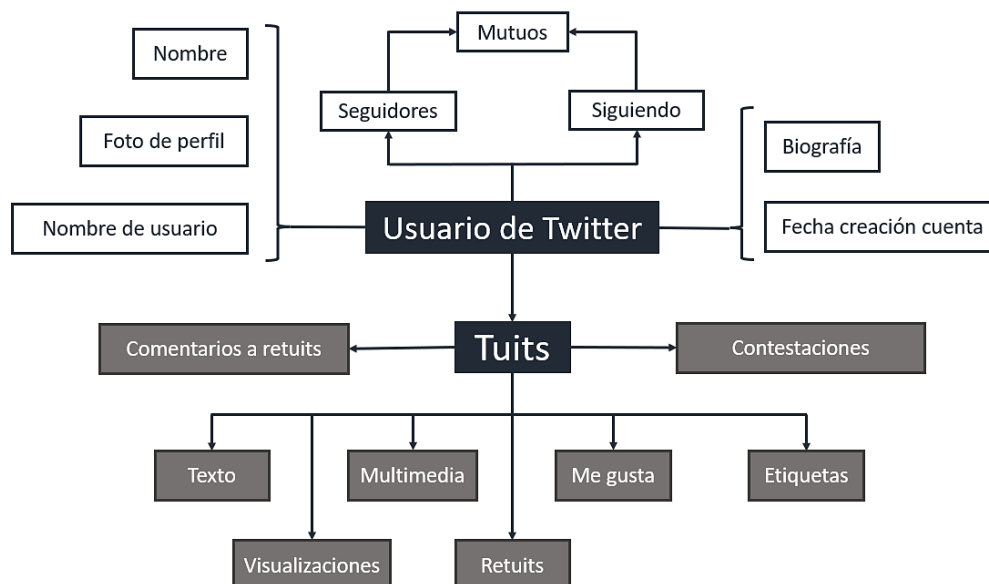


Figura 2-6 Estructura de información en Twitter. (Autoría propia)

2.2.3 API de Twitter

La API (*Application Programming Interface*, Interfaz de Programación de Aplicaciones) [33], es aquella interfaz mediante la cual se pueden comunicar entre sí los diferentes programas informáticos. El programa o aplicación, llama al denominado punto de conexión. Dichos puntos de conexión representan un enlace concreto entre dos programas. [34]

Twitter admite el acceso a los diferentes servicios que presta, con el objetivo de que los diferentes usuarios o entidades creen software integrado con la red social, mediante la extracción o publicación de datos en esta. De este modo, mediante el empleo de la API de Twitter, se puede extraer información intrínsecamente ligada a la aplicación, a los usuarios y a lo que estos han decidido compartir de manera pública. Del mismo modo, cada usuario podrá permitir acceso a su información privada, como sus mensajes directos, al desarrollador que previamente seleccione. [34]

Para tener acceso a la información proporcionada por dicha red social, es necesario el registro en una aplicación. El darse de alta en esta aplicación otorga de manera predeterminada tan solo acceso a información pública. De este modo, cabe la posibilidad de crear estadísticas, estudiar las tendencias temáticas, explorar diferentes usuarios en busca de información útil para propósitos de inteligencia, etc. por parte de terceros. La API de esta red social incluye diferentes funcionalidades que podemos englobar en los siguientes grupos: [34]

- Cuentas y usuarios: Facilitan a los desarrolladores la administración del perfil y configuración de una cuenta, así como silenciar/bloquear diferentes usuarios o tuiteros. Una de las opciones ofrecidas por esta interfaz es la consecución de información acerca de la actividad de una cuenta.
- Tuits y respuestas: Cualquier tuit público se encuentra a disposición de cualquier desarrollador. Estos tienen la capacidad de acceder a ellos mediante la búsqueda de palabras, frases o etiquetas clave. Esta API también permite la publicación de tuits o respuestas.

- Mensajes directos: Como se ha expuesto, tan solo tendrán acceso a ellos los desarrolladores a los que se les ha otorgado de manera explícita un permiso.
- Anuncios: Permite a los diferentes desarrolladores que trabajan mediante estas interfaces la creación y administración de campañas y anuncios mediante la red social. Esta permite a los programadores la obtención de inteligencia mediante los diferentes tuis y estadísticas de cada cuenta para llevar a cabo un seguimiento de las diferentes campañas publicitarias.
- Herramientas y SDK de editor: También conocidas como las Twitter *Cards*, cuya motivación radica en la integración de múltiples contenidos de la red social Twitter en diferentes páginas web mediante el uso de una tarjeta. Mediante un botón implementado en cualquier página, se permite el acceso a contenido de la aplicación.

2.3 Empleo de herramientas OSINT en Twitter

Twitter, como se ha explicado previamente, es una de las redes sociales que cuenta en la actualidad, con mayor número de usuarios. Por esta razón, es imprescindible conocer el método y las herramientas para realizar una investigación de fuentes abiertas en Twitter.

Una primera herramienta a considerar es la denominada “Temas” (véase Figura 2-7) cuya función principal es hacer una distinción de aquellas categorías tratadas mediante la publicación de tuits o las diferentes conversaciones y discusiones entre usuarios acerca de alguna noticia en concreto, a través de *Machine Learning* (aprendizaje automático, véase apartado 2.5.1 para más detalle). Con el empleo de una serie de algoritmos, esta herramienta hace un estudio constante de las conversaciones, retuits, menciones, me gusta, etc., que derivan de cada publicación y va creando los conocidos como temas. [35] Estos temas se convierten pues, en una facilidad que la red social ofrece a cada usuario permitiéndole elegir aquellos que sean de mayor interés para él; de tal manera que, le irán apareciendo recomendaciones de discusiones o simplemente conversaciones y tuits que estén tratando uno de los tópicos que previamente ha seleccionado.

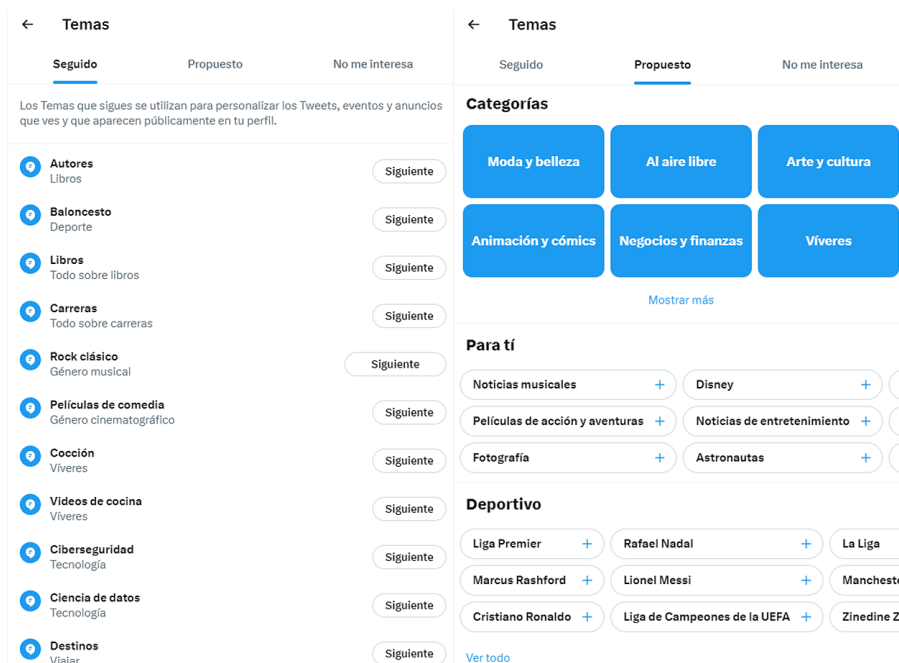


Figura 2-7 Herramienta "Temas" en Twitter. (Autoría propia)

Esta herramienta guarda cierta similitud con la que se pretende desarrollar en este proyecto, sin embargo, ambas difieren en los fines que persiguen. En primer lugar, el objetivo es la creación de un sistema de clasificación inteligente de usuarios, y no la aparición de tuits, focalizados en un tema en específico, en la cronología de inicio de cada usuario. A su vez, la herramienta a llevar a cabo no estará

integrada en la propia red social lo que permitirá facilitar su adaptación futura a otro tipo de redes sociales.

Otra herramienta eficaz de OSINT que Twitter ofrece a los diferentes usuarios para realizar las dispareas indagaciones de manera general, es la conocida como “Búsqueda avanzada” (véase Figura 2-8). Mediante esta herramienta, se pueden realizar múltiples filtraciones en distintos parámetros en función del tuit, usuario, etiqueta, fecha de publicación, etc. que se pretenda encontrar. Si bien es cierto que el empleo de esta herramienta no llega a ser del todo convincente, debido a que se obtienen resultados muy genéricos y con demasiado ruido.

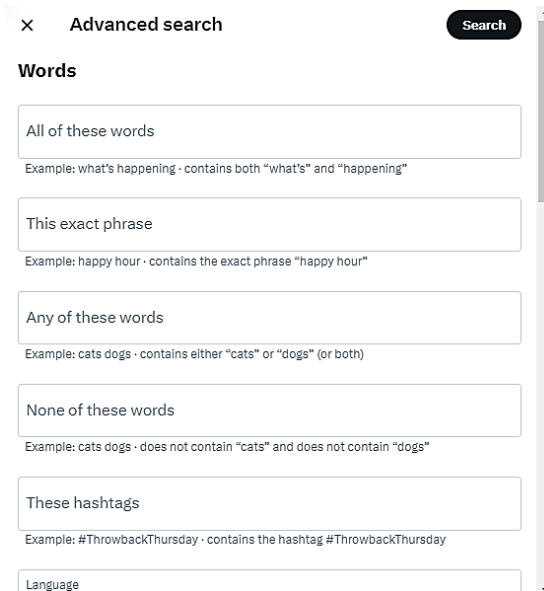
The image shows the 'Advanced search' interface on Twitter. At the top, there is a close button (X) and the title 'Advanced search'. A 'Search' button is located in the top right corner. Below the title, the section 'Words' is highlighted. There are five input fields with the following labels and examples: 1. 'All of these words' with the example 'what's happening · contains both "what's" and "happening"'; 2. 'This exact phrase' with the example 'happy hour · contains the exact phrase "happy hour"'; 3. 'Any of these words' with the example 'cats dogs · contains either "cats" or "dogs" (or both)'; 4. 'None of these words' with the example 'cats dogs · does not contain "cats" and does not contain "dogs"'; 5. 'These hashtags' with the example '#ThrowbackThursday · contains the hashtag #ThrowbackThursday'. At the bottom, there is a 'Language' field.

Figura 2-8 Búsqueda avanzada en Twitter. (Autoría propia)

Si el objetivo del proyecto fuese obtener un filtrado de cuentas y la relación que mantienen entre ellas, la realización de un SNA (*Social Network Analysis*, análisis de una red social) podría ser de gran utilidad. Se obtienen grafos muy visuales (véase Figura 2-9) que permiten conocer patrones y anomalías en los usuarios correlacionados por diferentes motivos. Estos diagramas permiten conocer cuáles son las cuentas que poseen mayores interacciones en la plataforma. Para ello, existen diferentes métricas como, por ejemplo: el *engagement* de un usuario, mide la interacción existente entre este usuario y sus respectivos seguidores empleando las diez o doce últimas publicaciones; la visibilidad de tus publicaciones, la tasa de interacción e impresiones, etc. [36] Cabe destacar la existencia de una herramienta denominada Twitter Analytics [37] que permite realizar un exhaustivo estudio y comparación de las estadísticas de cada usuario.

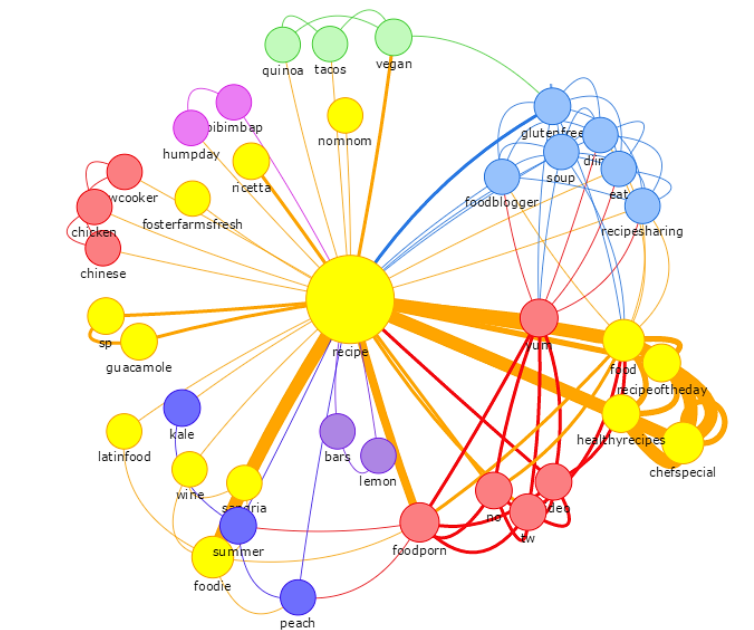


Figura 2-9 Grafo de SNA. [38]

La herramienta The Wayback Machine [39] permite, mediante la dirección de una cuenta de Twitter, ver capturas que tomó la propia plataforma en diferentes momentos y poder acceder asimismo a los tuits que ha borrado cada perfil. También se puede recurrir a la caché de Google, sin embargo, esta última no es tan eficaz, pues no guarda toda la información.

Cabe señalar, que, a pesar de que el nombre de usuario pueda variar a lo largo del tiempo, el ID se va a mantener siempre el mismo. Es por ello, que conocerlo nos va a aportar información muy valiosa.

2.3.1 Herramientas OSINT para Twitter

- TWINT (*Twitter Intelligence Tool*, Herramienta de Inteligencia de Twitter) [40]: Esta utilidad permite extraer información valiosa de la plataforma sin necesidad de usar la API. Existe la restricción de acceder a 3.200 tuits como máximo. Ofrece la posibilidad de filtrar mediante palabras, seguidores, etc. en base a los tuits que se desea obtener. Esta herramienta (véase Figura 2-10) está escrita en *Python*.

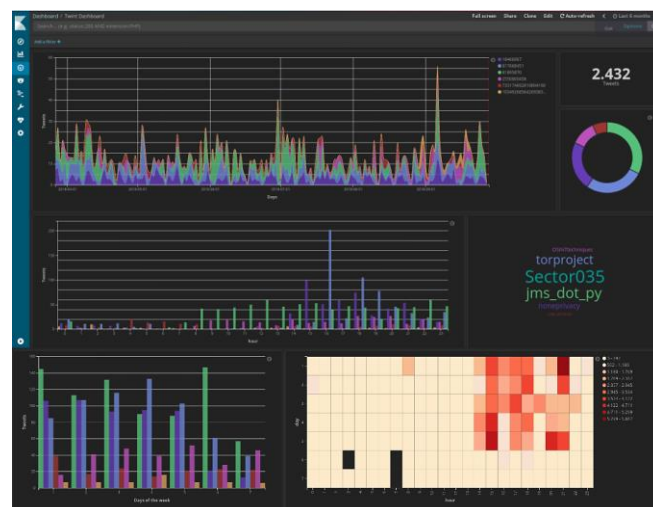


Figura 2-10 TWINT. [41]

- TWOSINT [42]: Contiene módulos con diferentes funcionalidades con el objetivo común de buscar tuits mediante el acceso a los seguidores y/o seguidos de un usuario, enlaces, etc. Esta es una herramienta de *Python* que utiliza la librería TWINT para agilizar y automatizar la exploración.
- TRUTH NEST [43]: Agiliza y explora, mediante el empleo de gráficas, estadísticas y diversidad de métricas, el análisis general de una cuenta en particular basado en sus tuits más destacables y populares, así como la frecuencia de uso, las etiquetas que ha empleado, la densidad de sus tuits, usuarios con los que más interactúa, etc. (Véase Figura 2-11)

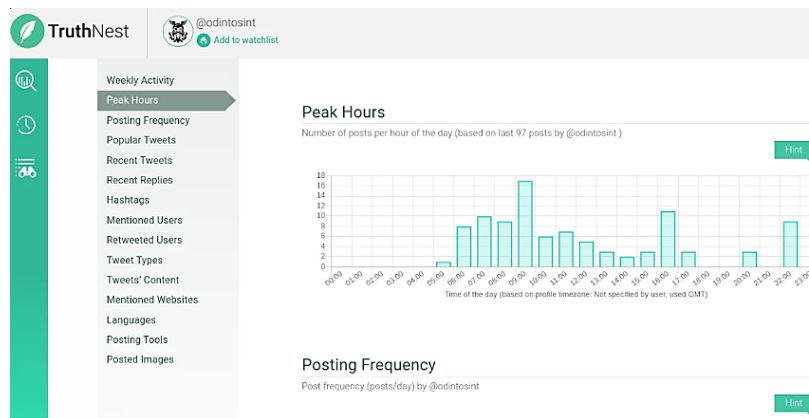


Figura 2-11 Truth Nest. [44]

- TINFOLEAK Y ACCOUNT ANALYSIS [45]: Esta herramienta de código abierto no precisa la utilización de *Python* para llevar a cabo las indagaciones; basta con la introducción del nombre de usuario y como consecuencia se obtienen múltiples datos acerca del mismo: actividad y dispositivos del usuario, frecuencia de uso, identidades digitales, ubicaciones principales del usuario, etc. [46] Dichos datos se envían en formato “html” al correo indicado, de tal manera que permita su posterior análisis para la generación de inteligencia.
- ALL MY TWEETS [47]: Muestra todos los tuits de un usuario, así como sus seguidores y línea del tiempo en una lista donde es posible filtrarlos. Es principalmente empleada para encontrar tuits antiguos. Posee un potente procesador que muestra en tan solo una página los últimos 3200 tuits publicados, en orden cronológico por dicha cuenta.
- FOLLOWERWONK [48]: Está enfocada principalmente a realizar un estudio acerca de los seguidores de diferentes cuentas y los mutuos entre ellas. Analiza las biografías de los diferentes usuarios con el objetivo de conectar a personas que puedan ser afines, así como analizar la competencia entre cuentas. Permite la realización de tareas de gestión, optimización y análisis. No precisa de un previo registro. [49]
- TWITTER AUDIT [50]: Consiste en una auditoría que permite la consecución del número aproximado de seguidores reales que tiene un determinado usuario mediante el análisis de una muestra de 5.000 de sus seguidores como máximo, comprobando cuántos de estos son reales en base al número de tuits, la fecha del último publicado, así como el ratio de seguidores y seguidos que el seguidor en cuestión tiene. Finalmente proporciona una puntuación de la calidad en función de las cifras obtenidas en dichos parámetros. Esta herramienta es capaz de bloquear, a su vez, a los seguidores falsos e inactivos si el usuario así lo requiere. Destaca también por la capacidad de análisis de las propias métricas de un usuario en cuanto a su contenido y alcance se refiere (véase Figura 2-12).

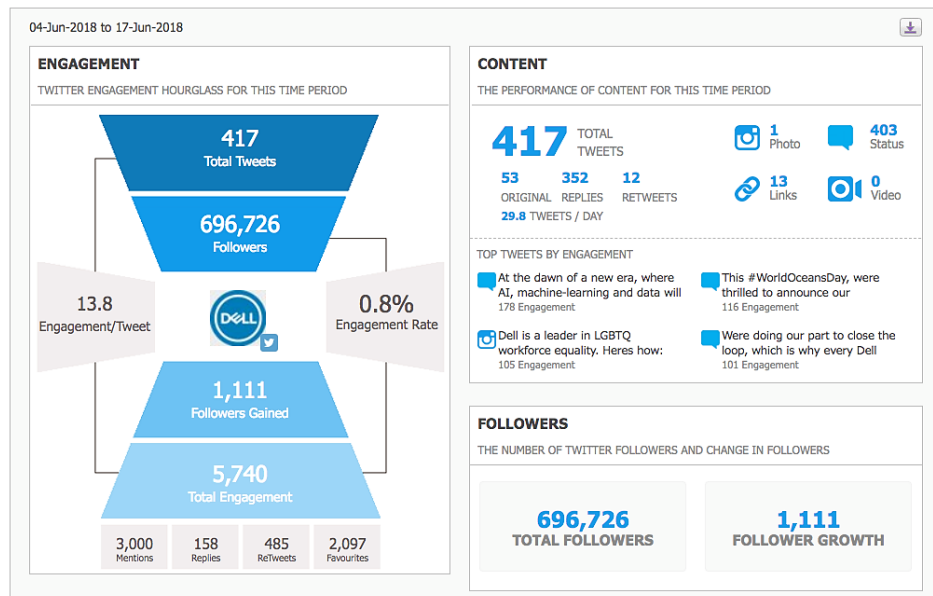


Figura 2-12 Twitter Audit. [51]

- **T-HOARDER:** [52] Herramienta destinada a la extracción de información de la red social Twitter mediante su API, a través de la monitorización y rastreo de tuits y usuarios, así como su capacidad de analizar y crear estadísticas sobre datos filtrados. Se trata de una herramienta gratuita de código abierto que, a su vez, también es capaz de crear grafos, *timelines* (líneas del tiempo), etc. Basa sus estudios en el seguimiento de ciertos parámetros determinantes para esta red social como etiquetas, conversaciones, menciones, etc. que tengan cierta relevancia a lo largo de un período prolongado de tiempo.

2.4 Detección de comunidades en Twitter

Las comunidades de Twitter no dejan de ser una funcionalidad que concilia la interacción entre usuarios de la plataforma que destaquen por tener intereses concretos y similares. Newman y Girvan definen comunidad de la siguiente manera: “Una comunidad es la división de una red en grupos dentro de los cuales las conexiones son densas, pero entre ellos escasas” [53]. El objetivo final de este instrumento radica en fomentar conversaciones e intercambio de ideas entre los diferentes usuarios. [54]

Debido al exponencial crecimiento de las redes sociales y el creciente interés que estas han demostrado desarrollar durante los últimos años, se ha comenzado a considerar relevante la detección de diferentes comunidades de usuarios que existen en estas plataformas. Por esta razón se han desarrollado diferentes algoritmos para solventar dicha cuestión. Muchos desarrolladores y programadores consideran este dilema estático, mientras que otros defienden su dinamismo, es decir, considera sus cambios respecto al tiempo. Sin embargo, lo que sí comparten es la importancia de las consecuencias derivadas de estas herramientas de detección de comunidades en diferentes ámbitos como la política, economía, el análisis de mercado, defensa nacional, etc.

Este problema radica en el modelado y optimización de un problema de partición de un grafo, la mayor parte de las veces con varios objetivos o condicionantes a la hora de seleccionar la comunidad. [55] El reconocimiento de estas particiones permite la comprensión de las interconexiones y de la red en sí. [56]

Si se parte de una red o grafo, la detección de comunidades se basará en la optimización de particiones que cumplan con los requisitos fijados previamente para dicha comunidad, como se puede

observar en la Figura 2-13. En el grafo mostrado, son identificables tres subgrafos o comunidades, en base a las interconexiones existentes entre los diferentes nodos.

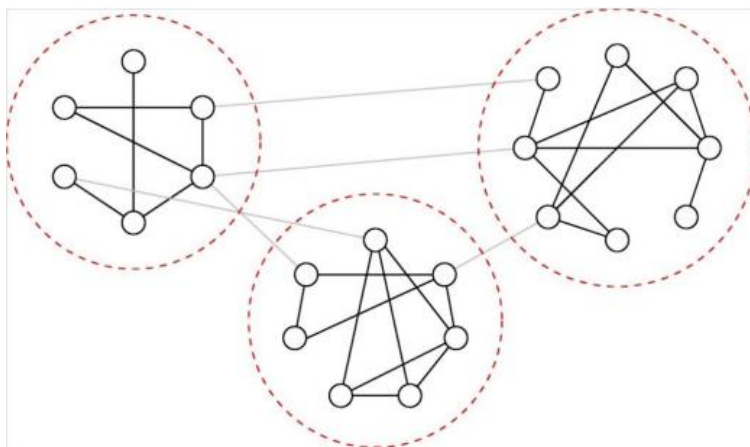


Figura 2-13 Descripción gráfica del concepto de Girvan y Newman. [53]

Con el objetivo de determinar las bases que han de cumplir los diferentes nodos para poder formar parte de una comunidad existen unos diversos criterios y herramientas, como, por ejemplo, la distancia euclídea, la distancia Manhattan o la similitud del coseno. [53] Existen diferentes métodos que permiten la detección de comunidades:

- **Métodos jerárquicos:** Para su empleo, se establecen las relaciones de los diferentes usuarios mediante una estructura de árbol denominada dendrograma (véase Figura 2-14). El equivalente a las hojas quedaría representado por los diferentes nodos, mientras que las ramas harían referencia a las relaciones entre ellos. La creación de comunidades viene determinada por la medida del grado de similitud que se pretende que tengan los nodos entre sí. Para la agrupación se pueden usar métodos divisivos: se conforman las comunidades mediante la eliminación de aquellas aristas que posean baja interacción; o a través de métodos aglomerativos. Estos últimos funcionan de manera inversa a los divisivos, pues conjuntos de nodos (que sean más similares entre sí) se van uniendo de manera iterativa agrupando comunidades con gran interacción. [53]

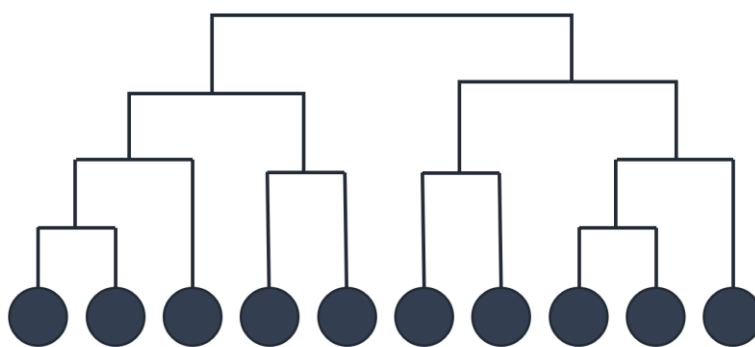


Figura 2-14 Dendrograma. (Autoría propia)

- **Métodos particionales:** Este modelo se basa en la medida de distancia, la cual nace de la similitud entre dos puntos o nodos. El fin último es crear particiones en las que se minimice la función del coste de las distancias entre nodos. Este método también engloba aquellas particiones que se realizan con el objetivo de minimizar el número de aristas entre diferentes comunidades.
- **Métodos para detección de comunidades con solapamientos:** Cabe destacar, que un nodo puede pertenecer a más de una comunidad de manera simultánea. En la siguiente

imagen (Figura 2-15) se puede observar este hecho. Es por ello, que se han desarrollado diferentes técnicas que permitan la localización de estas comunidades independientemente a la pertenencia a más de una de ellas. Son ejemplo de estas técnicas: el método *Clique Percolation Method* (CPM, Método de Percolación de Cliques) [53], el cual parte del grafo y mediante las diferentes aristas y uniones de nodos calcula las diferentes comunidades; o el método CPM secuencial, cuyo funcionamiento es opuesto al anterior, puesto que busca cierto número de comunidades mediante la inserción de aristas.

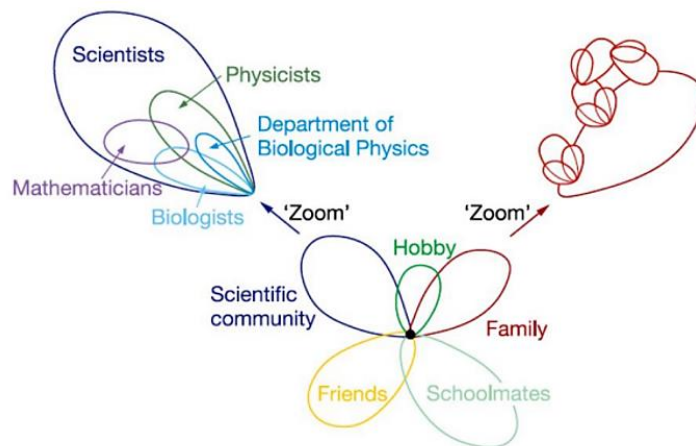


Figura 2-15 Gráfico de comunidades solapadas. [53]

- **Métodos espectrales:** Se basan en un algoritmo compuesto por ciertos datos, de los que se estudia sus similitudes mediante el empleo de una matriz simétrica y no negativa que permite construir un grafo del cual se extrae el componente similar entre los diferentes nodos o usuarios. Entre estos algoritmos se encuentran: algoritmo NG, Jordan y Weiss; y el algoritmo Shi y Malik. [53]

Durante el proceso de detección de comunidades, independientemente del método empleado, uno de los factores más importantes a tener en cuenta, es la verificación de la calidad de los resultados obtenidos. Cabe destacar la importancia de determinar cuán vulnerable es la comunidad obtenida al ruido, a las variaciones que con el tiempo se producen, etc.

2.5 Inteligencia artificial

Jeff Hawkins definía el término inteligencia en su libro *“On Intelligence”* (“Sobre la inteligencia”) de este modo: *“Capacidad para recordar y anticipar pautas en el mundo incluyendo lenguaje, matemáticas, propiedades físicas de los objetos y simulaciones sociales”* [57]. Al contrario que Turing, quien proponía como test de inteligencia la capacidad de hacerse pasar por una persona ante un observador externo; Hawkins asegura que la inteligencia no guarda relación con la conducta sino con la capacidad de anticipación. [58]

La primera aparición del término inteligencia artificial se remonta a la década de 1950 con John McCarthy, miembro del departamento de Ciencias Informáticas de la Universidad de Stanford durante una conferencia en Dartmouth. Si bien es cierto que sería Alan Turing el considerado padre de esta rama de la ciencia. Este último lanzaba la siguiente pregunta: *“¿Pueden las máquinas pensar?”* a través de un artículo en la revista *Mind*. Todo ello trajo consigo una revolución en el ámbito computacional de la automatización, sistemas de toma de decisiones y búsquedas inteligentes hasta alcanzar los niveles que actualmente se conocen. [59]

La evolución de las técnicas de inteligencia artificial [60] puede clasificarse en tres grandes etapas. La primera de ellas (1950-1970) en la cual se comenzaron desarrollando una serie de algoritmos y métodos con el objetivo de resolver problemas, en unos inicios relativamente sencillos. La segunda

(1970-1990) en la cual se desarrollaron sistemas de aprendizaje expertos también denominados sistemas de conocimiento. Fue en este período en el que inversores privados comenzaron sus inversiones en este ámbito. Y, finalmente la tercera (1990-actualidad) en la que se está llevando a cabo un exhaustivo desarrollo del aprendizaje automático y aprendizaje profundo.

Se define inteligencia artificial (IA) [59] como la capacidad que posee una máquina para exhibir un conjunto de habilidades y realizar tareas comúnmente asociadas con la cognición humana tales como la posibilidad de razonar, aprender, ser creativo, etc. Este tipo de inteligencia perteneciente a la rama de ciencias de la computación busca dotar a dichas máquinas con una capacidad de razonamiento y actuación que simulen las del propio ser humano. Su consecución se basa en tres parámetros fundamentalmente [61]: los sistemas computacionales, los datos y gestión de estos y finalmente los algoritmos avanzados de inteligencia artificial que permitan lograr la adquisición de esas capacidades mediante aprendizaje basado en la experiencia y análisis de patrones o características del conjunto de datos a estudiar. El criterio más aceptado en la actualidad para definir este concepto es el expuesto en el libro “*Artificial Intelligence: A Modern Approach*” por Peter Norvig y Stuart Russell [62], informáticos que aseguraban que la inteligencia artificial podía clasificarse según cuatro criterios: sistemas que actúan como humanos, sistemas que actúan racionalmente, sistemas que piensan como humanos y sistemas que piensan racionalmente. Básicamente estos criterios pueden dividirse en dos principios: actuar y pensar.

Esta rama de la ciencia nace de la necesidad de procesar la cantidad ingente de datos que ha ido aumentando exponencialmente en estos últimos años debido principalmente a la cuarta revolución industrial y al desarrollo incesante que las nuevas tecnologías están teniendo. Este aumento de información trasvasa los límites que el ser humano es capaz de tratar, analizar y procesar. Es por ello que, este tipo de inteligencia y aprendizaje es fundamental para automatizar la toma de decisiones en muchos ámbitos que el ser humano no es capaz de abarcar.

Cabe destacar a su vez, que la mayoría de las aplicaciones [63] que tiene esta tecnología se fundamentan hoy en día en el empleo de aprendizaje automático y profundo (explicado más adelante 2.5.1) así como en la robótica, el procesamiento del lenguaje natural, razonamiento y resolución de problemas y planificación, entre otros. Gracias a esto, existe la posibilidad de entrenar a las máquinas para llevar a cabo tareas complejas y específicas sin ser previamente programadas para ello, mediante un aprendizaje por experiencia.

Como bien se ha explicado previamente, una de las principales características de la inteligencia artificial es el aprendizaje constante y de manera automática por parte de una máquina. Es aquí donde nace la rama de esta clase de inteligencia denominada *Machine Learning*.

2.5.1 *Machine Learning*

Se pueden clasificar los métodos de inteligencia artificial en dos tipos: el desarrollo dirigido por modelos (*Model-Driven*, MD) o el dirigido por datos (*Data-Driven*, DD). Los MD precisan instrucciones para poder concluir una solución, mientras que los DD emplean el conjunto de datos y sus soluciones para ser capaces de llevar a cabo un entrenamiento mediante el cual obtener las reglas o patrones para que pueda llegar a una solución la próxima vez que se le proporcionen una serie de datos. Esto es conocido como *Machine Learning* (véase Figura 2-16).

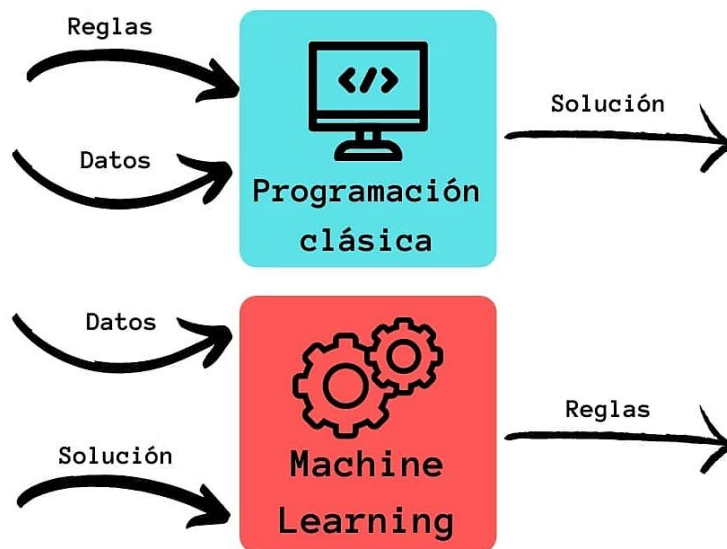


Figura 2-16 Machine Learning. [60]

Machine Learning [64], o aprendizaje automático, es una disciplina perteneciente a la inteligencia artificial, mediante la cual se dota a los computadores de una capacidad que les permite procesar una cantidad ingente de datos con el fin de encontrar patrones de comportamiento para poder elaborar predicciones, es decir, llevar a cabo un análisis predictivo. Este modo de aprendizaje permite a los ordenadores poder llevar a cabo un aprendizaje de manera autónoma, sin tener que ser programados expresamente para ello. Si bien es cierto, que este tipo de aprendizaje presenta una serie de limitaciones y restricciones en los algoritmos que permiten entrenar dichas herramientas, pues la mayor parte de las veces precisan de conjuntos de datos de entrenamiento que previamente han tenido que ser etiquetados por seres humanos (aprendizaje supervisado).

Fue Arthur Samuel, pionero en el ámbito de los videojuegos y empleado de IBM, la primera persona en emplear este término en 1959 [65] en un proyecto mediante el cual consiguió que un ordenador venciese a un ser humano jugando al juego de las damas. Sin embargo, no fue hasta los años 90, en los que Internet comenzó a avanzar y desarrollarse a pasos agigantados y junto con él la cantidad de datos accesibles, cuando el aprendizaje automático comenzó a ganar relevancia hasta convertirse en una parte imprescindible de Big Data [66] e IoT (*Internet of Things*, Internet de las cosas) [67] entre otros. Actualmente es reconocido y empleado a diario a nivel global prácticamente de manera inconsciente en hábitos rutinarios tales como el empleo del reconocimiento facial en dispositivos, la detección de spam en el correo electrónico, etc.

Este aprendizaje automático está evolucionando a *Deep Learning* [65], o aprendizaje profundo, con el que se pretende que no sea necesaria la intervención humana previa, o que esta sea mínima. Una de las bases de este tipo de aprendizaje es el empleo de estructuras lógicas similares a la microarquitectura del sistema nervioso (redes neuronales) del ser humano, en las que existen capas de unidades de proceso, neuronas artificiales, enfocadas a detectar características presentes y específicas de objetos.

2.5.2 Aplicaciones del Machine Learning

El empleo de técnicas de aprendizaje automatizado se ha normalizado a nivel global. Actualmente resulta esencial en muchas de las actividades y ámbitos presentes en el día a día de las personas. Cabe destacar su empleo principalmente en los siguientes ámbitos [65]:

- Ciencias de la salud: Es una de las principales herramientas utilizadas en las diferentes investigaciones de enfermedades y tratamientos, pues presta la capacidad de realizar análisis y

predicciones de manera precisa a partir de bases de datos, identificando patrones que en la mayoría de los casos resultan de vital importancia para dichos proyectos.

- Industria financiera: Crucial para la detección de prácticas fraudulentas, percepción de riesgos en inversión, así como la mejora que proporciona en la toma de decisiones económicas en tiempo real. Los bancos basan su funcionamiento en algoritmos de aprendizaje máquina para analizar el comportamiento de clientes y detectar operaciones sospechosas o transacciones fraudulentas.
- Marketing: Emplea estas herramientas para encontrar patrones de búsqueda, en función de historiales de compras y preferencias, analizarlos y realizar predicciones cuyo objeto es personalizar las ofertas, anuncios y mensajes que se les proporciona a los clientes y, de este modo, mejorar su retención y fidelización.
- Industria logística: En este ámbito, es empleado el aprendizaje automático con fines de optimización de la gestión de la cadena de suministro y predicción de la demanda, así como para reducir costes, mejorar la eficiencia y la precisión de las empresas. Es empleado a su vez para predecir las rutas óptimas para ir de un sitio a otro, en base al tráfico, posibles atascos, consumo de combustible, etc.
- Procesamiento de lenguaje natural (PLN) [64]: Se ha generalizado su uso para comprensión del lenguaje humano en asistentes virtuales empleados a diario tales como Alexa o Siri, quienes en tan solo segundo son capaces de traducir idiomas, reconocer la voz del locutor y analizarla. Cabe destacar también el empleo de aprendizaje automático en traductores de jergas complejas a un lenguaje más comprensible y sencillo.

2.5.3 Métodos de Machine Learning

Dentro de este aprendizaje automático existen cuatro grandes métodos [65] en base al cúmulo de operaciones ejecutadas sistemáticamente o algoritmos empleados:

1. Aprendizaje supervisado: Consiste en, mediante diferentes algoritmos, deducir en base a conjuntos de datos, a los cuales se les ha asignado una etiqueta previamente, la etiqueta que se le asignaría a un conjunto de datos de entrada. Cabe destacar, que el algoritmo se entrena mediante el empleo del histórico de datos ya etiquetados, mencionados al inicio, siendo el segundo conjunto para proceder con la prueba y evaluación del modelo creado mediante la predicción del valor de salida que se le asignaría. La principal diferencia con el resto de los métodos radica en la necesidad de supervisión humana. Este tipo de aprendizaje es el que se empleará para la construcción de un clasificador entrenado como más adelante se mostrará (Apartado 3.6).
2. Aprendizaje por refuerzo (*Reinforcement Learning*): Método mediante el cual un algoritmo basa su aprendizaje en el avance de otro algoritmo, con el método prueba-error. De este modo se obtienen óptimos resultados en base a muestras y experiencias.
3. Aprendizaje no supervisado: La principal diferencia con el método descrito previamente, radica en el empleo de datos de entrenamiento no etiquetados, es la propia máquina la que realiza un análisis de estos obteniendo patrones sin necesidad de intervención humana. El ejemplo más habitual es el agrupamiento (*clustering*) de registros en un conjunto de datos en base a su similitud.
4. Aprendizaje semisupervisado: Combina ambos métodos mediante el empleo de ambas clases de datos, etiquetados y no. Cabe destacar que el número de datos etiquetados suele ser más reducido que en el caso del supervisado.

2.5.4 Aprendizaje automático supervisado

Cabe profundizar en el aprendizaje automático supervisado, en el cual se encuentran dos principales métodos de aprendizaje: clasificación (que se va a emplear en el presente proyecto), y regresión [68]. La diferencia principal entre ellos radica en que los métodos de clasificación son entrenados para clasificar valores discretos y tienen como función principal predecir a qué clase

pertenece un conjunto de datos de entrada. Sin embargo, la regresión entrena un algoritmo con el fin de predecir valores continuos.

Uno de los factores más importantes a tener en cuenta al seleccionar como método de trabajo el aprendizaje automático supervisado, es el seguimiento de un procedimiento estipulado que cuenta con los siguientes pasos (véase Figura 2-17):

1. Realizar una elección a conciencia del conjunto de datos que se emplearán en el entrenamiento determinando la naturaleza de los mismos: si es una letra, una cadena de texto, un dígito, etc.
2. Hacer una exhaustiva limpieza del conjunto de datos para evitar información irrelevante o que pueda afectar negativamente a los posteriores pasos.
3. Elección del algoritmo a emplear en función del uso que se le pretenda dar, de la naturaleza de la información de entrada, así como del tiempo de respuesta, precisión en la predicción y transparencia que se pretende obtener.
4. Entrenamiento del modelo obtenido mediante la repetida ejecución de este sobre los datos de entrenamiento (supervisión).
5. Proceder a la evaluación del modelo ya entrenado una vez que se haya obtenido un ajuste válido con un nuevo conjunto de datos.

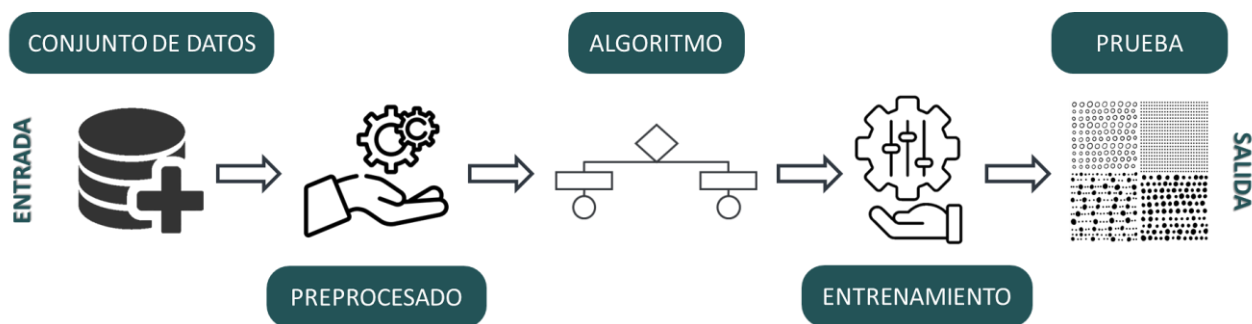


Figura 2-17 Metodología de aprendizaje automático supervisado. (Autoría propia)

Como bien se ha señalado, existen diferentes algoritmos pertenecientes a este tipo de aprendizaje supervisado. Entre los principales se encuentra:

- **Árbol de decisión:** Como bien se puede entender por su propio nombre, consiste en un modelo de predicción estructurado simulando un árbol en el que se parte del nodo raíz. Los nodos representan una serie de preguntas siendo las ramas, que de ellos salen, la representación de las respuestas (sí o no) con el fin de llegar a una decisión final [69]. Es comúnmente empleado en tareas y resolución de problemas que no precisen mucha complejidad, debido a la sencillez que presenta a la hora de clasificar. La representación de este algoritmo es semejante a la de un diagrama de flujo (véase Figura 2-18).

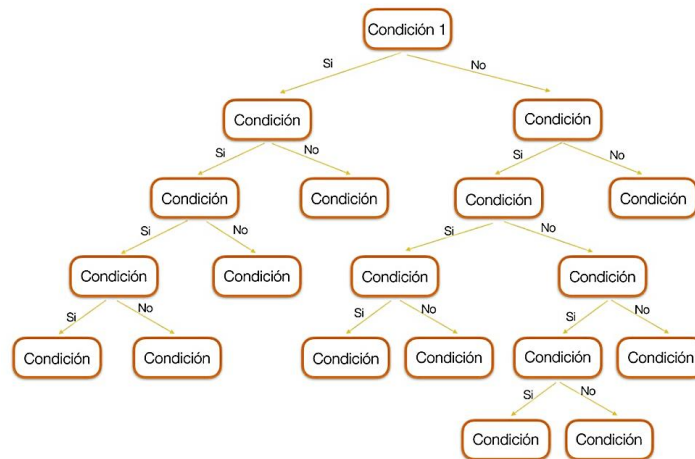


Figura 2-18 Ejemplo del algoritmo de árbol de decisión. [70]

- *Random Forest*: Este método [70] será el empleado para la realización del clasificador en el presente trabajo. Se basa en un aprendizaje automático polivalente, pues es capaz de llevar a cabo tareas de clasificación o regresión, así como reducción dimensional tratando datos atípicos o perdidos. Nace de la necesidad de paliar las limitaciones del algoritmo de árbol de decisión, pues una vez ya entrenados, estos árboles tienden a perder precisión con otro conjunto de datos. Se fundamenta en la ejecución de estos árboles de manera combinada y simultáneamente. Llevan a cabo una clasificación mediante la cual, cada árbol llega a una conclusión, siendo la final la que más se reproduzca, es decir, la que más votos tenga (véase Figura 2-19). Una de las principales ventajas que presenta este algoritmo es la capacidad de trabajar con miles de variables de entrada en un conjunto de datos e identificar y cuantificar cuáles tienen mayor relevancia. Este algoritmo, perteneciente a la familia de algoritmos denominada de ensamble, puede ser empleado tanto para clasificación (*RandomForestClassifier*), donde se tomará como decisión final el voto mayoritario, como para regresión (*RandomForestRegressor*), en el cual se empleará el valor medio de las predicciones de todos los estimadores [71]. El funcionamiento de este algoritmo radica en la creación de un número aleatorio de árboles de decisión mediante una selección también aleatoria de conjuntos de datos de entrenamiento provenientes de un corpus. Se creará un árbol por cada uno de los conjuntos en función de unas características previamente establecidas, denominando a este procedimiento *bagging*. Se reservan del corpus una serie de datos que serán empleados posteriormente al entrenamiento para validar el modelo. [71]

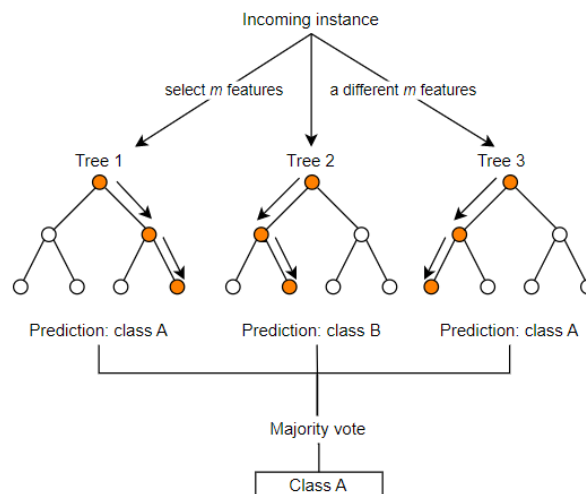


Figura 2-19 Representación gráfica del algoritmo *Random Forest* para clasificación. [72]

- Clasificación de Naïve Bayes: Basa su funcionamiento en el teorema de Bayes, en el cual se parte de la independencia de las clases. Su simplicidad le convierte en un sencillo y recurrente modelo de clasificación. La principal característica de este algoritmo es la presumible ingenuidad de la que parte, pues asume que las diferentes características de los datos son independientes entre sí, esto implica que la posesión de una característica no indica ni afecta a la presencia de otra. Dentro de este método existen distintas técnicas: clasificador de Bernouilli, multinominal y Gaussiano. [73]
- *Support Vector Machines* (Máquina de Vector Soporte, SVM): Este algoritmo desarrollado por el ruso Vladimir Vapnik destaca por su flexibilidad de empleo tanto en clasificación como regresión; no obstante, su empleo en clasificación está mucho más generalizado. El fin principal de este método es la obtención de un hiperplano que separe y distinga de la manera más precisa dos clases distintas de un conjunto de datos, véase maximizando el margen (anchura representada mediante paralelas al hiperplano en la cual no se encuentran datos pertenecientes a ninguna de las clases) entre ambas características. En el ejemplo de la Figura 2-20, se puede observar el hiperplano y el margen existente entre las supuestas clases (representadas por los símbolos '+' y '-'), así como los denominados vectores de soporte (puntos límites pertenecientes a cada clase a partir de los cuales se calcula el hiperplano y el margen). [73]

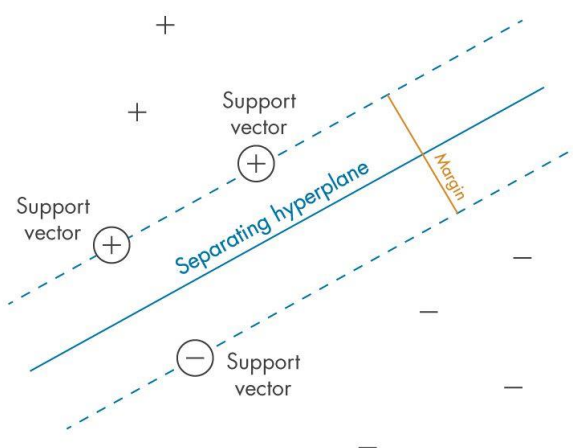


Figura 2-20 Ejemplo del cálculo del hiperplano y el margen en SVM. [74]

- Regresión logística: Se trata de un algoritmo estadístico empleado con el fin de solventar problemas de clasificación binaria calculando la probabilidad de cierto suceso. Su entrenamiento radica en maximizar, mediante métodos de optimización, la probabilidad de que los datos sean clasificados correctamente. Es empleado en herramientas destinadas a las ciencias de la salud y sociales con el objeto de predecir el valor de una variable categórica en función de variables independientes. [73]
- *K-nearest neighbor*, KNN: Este algoritmo, traducido como K-vecino más próximo, es un algoritmo cuya función es clasificar puntos de datos en base a su proximidad y similitud a otros datos. Emplea la distancia euclidiana para determinar la distancia que separa el punto a analizar del resto ya categorizados. Estas distancias son ordenadas de menor a mayor seleccionando las k menores y asignándole al punto analizado una categoría en función de la más frecuente o la promedio. Su funcionamiento con una baja cantidad de datos es óptimo pues destaca por su sencillez y su alta velocidad. Si bien es cierto que este buen funcionamiento es inversamente proporcional a la suma de información que tenga como entrada, por lo que sus características irán en decadencia a medida que aumenten los datos haciéndolo menos atractivo. En la Figura 2-21 se muestra un ejemplo del algoritmo para K=4 en el cual se hayan las mínimas distancias entre el punto a obtener la categoría y el resto ya

categorizados seleccionando los 4 más cercanos. Se concluye que, puesto que tres de ellos pertenecen a la categoría B, el objeto en estudio pertenece también a esta clase. [73]

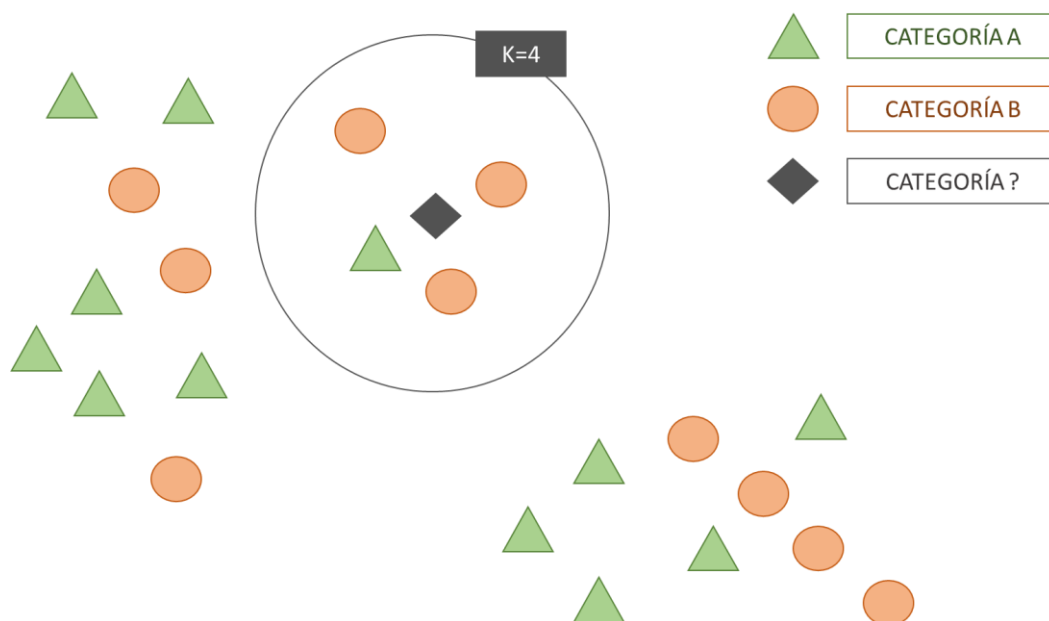


Figura 2-21 Algoritmo KNN con $K=4$. (Autoría propia)

2.5.5 Modelos de lenguaje

Uno de los mecanismos de inteligencia artificial más desarrollados y entrenados en el momento es el de los modelos de lenguaje de gran tamaño, también conocidos como LLM (*Large Language Model*). Estos modelos de lenguaje de gran tamaño han impulsado un elevado avance en el desarrollo de la inteligencia artificial y son empleados con fines relacionados con el procesamiento del lenguaje natural o NLP (*Natural Language Processing*), un ejemplo es la traducción de texto mediante su previa comprensión y sintetización. Otras de sus funcionalidades se encuentran en la generación de texto, así como la posible respuesta a preguntas realizadas en diferentes idiomas, realizadas por un usuario de dichas herramientas. Estas técnicas de procesamiento de lenguaje natural están íntimamente ligadas al aprendizaje profundo (expuesto a lo largo de este epígrafe), pues basan su funcionamiento en este tipo de métodos de aprendizaje. [75]

Dichos modelos pueden ser entrenados con diversos tipos de texto: artículos de investigación, conversaciones, novelas, diversas publicaciones, historiales de búsqueda, etc. Normalmente se emplean técnicas de aprendizaje auto supervisado (*self-supervised*) como puede ser la predicción de la siguiente palabra en un texto. Esto permite utilizar grandes volúmenes de datos (por ejemplo, datos web) para el entrenamiento, lo que redundará en modelos grandes con millones de parámetros, dando lugar al nombre que recibe. Destacan por ser capaces de obtener una alta precisión en las salidas de sus algoritmos y metodología, así como por ser capaces de llevar a cabo complejas tareas muchas veces difíciles de realizar por humanos en cuanto a procesamiento de lenguaje natural se refiere. [75]

Entre los ejemplos más afamados y destacables de los modelos de lenguaje LLM se encuentran [75]:

- Traducción de texto prácticamente instantánea y automática de un idioma a otro.
- Empleo de *cookies* y algoritmos que emplean el historial de búsqueda de un usuario para encontrar un patrón en sus búsquedas y así poder proporcionarle resultados personalizados.
- Algoritmos de análisis de sentimiento que permitan discernir las opiniones positivas de las negativas acerca de un producto o un servicio.
- Sistemas de redacción automática de un texto basado en unas indicaciones previas o pautas sobre el contenido a redactar.

- Desarrollo de sistemas de comunicación entre el usuario y la computadora también denominados *chatbots*. Dentro de estos encontramos el afamado ChatGPT [76] sobre el que se tratará a continuación.

2.5.5.1 OpenAI y ChatGPT-3.5

OpenAI es una organización fundada por un grupo de científicos e inversores, entre los que destaca Elon Musk, en el año 2015 y focalizada en el ámbito del desarrollo, investigación e implementación de inteligencia artificial. Esta defiende el acceso libre y para todos a este tipo de tecnología, desarrollando sistemas *opensource*, algunos de los cuales han comenzado recientemente a comercializarse. [77]

Se procede en el presente apartado a explicar en qué consiste la herramienta impulsada por OpenAI [78] denominada ChatGPT (*Chat Generative Pre-trained Transformer*) [76] pues, aparte de ser la revolucionaria interfaz que llevó a esta organización al mayor de sus éxitos, será de gran relevancia para llevar a cabo una comparación dentro del epígrafe de Prueba y validación (capítulo 4) con el sistema de detección desarrollado en el actual proyecto.

Se entiende por *chatbot*, el programa informático que, mediante el empleo de inteligencia artificial, tiene la capacidad de mantener una conversación a través de Internet con un internauta simulando y razonando como lo haría un interlocutor humano mediante el empleo de lenguaje natural. Es necesario remontarse a mediados de los años 60 para conocer los inicios de estos robots. Esta herramienta, desarrollada y entrenada por una red neuronal, tiene la capacidad de generar respuestas mediante el empleo de técnicas de *Machine Learning*. El Instituto de Tecnología de Massachusetts (MIT, *Massachusetts Institute of Technology*) fue el precursor de este tipo de tecnología mediante el desarrollo de “Eliza” en 1966 cuyo nombre derivó a “*chatter bot*”, y cuya misión era la creación de una interfaz mediante la cual poder conectar al ser humano con la máquina para llevar a cabo una conversación. No fue hasta 1972 que se desarrolló el que sería su sucesor, “*Parry*”, gracias a la Universidad de Stanford. Ambos fueron conectados para comunicarse mediante ARPANET. Tras estas primeras apariciones, llegó la conocida era de los *chatbots* en la cual se comenzaron a lanzar robots que imitaban la voz humana (como “*Jabberwacky*”), aquellos que empleaban interfaces de usuario e imitaban respuestas de psicólogo (“*Dr. Sbaits*”), el famoso robot llamado “*Alice*”, etc. Todo ello ha ido evolucionando y mejorando con los años hasta llegar al que es considerado el *chatbot* más importante en la actualidad: ChatGPT. [79]

La versión 3.5 de ChatGPT, lanzada en noviembre del año 2022, es uno de los modelos de procesamiento de lenguaje natural más desarrollados y capaces existentes en la actualidad. La principal finalidad de esta herramienta es la creación de *chatbots* o agentes conversacionales (véase interfaz del chat en la Figura 2-22), es decir, responder a preguntas realizadas por un usuario en función a una base de datos (cuenta con más de 175 mil millones de parámetros), previamente almacenados.

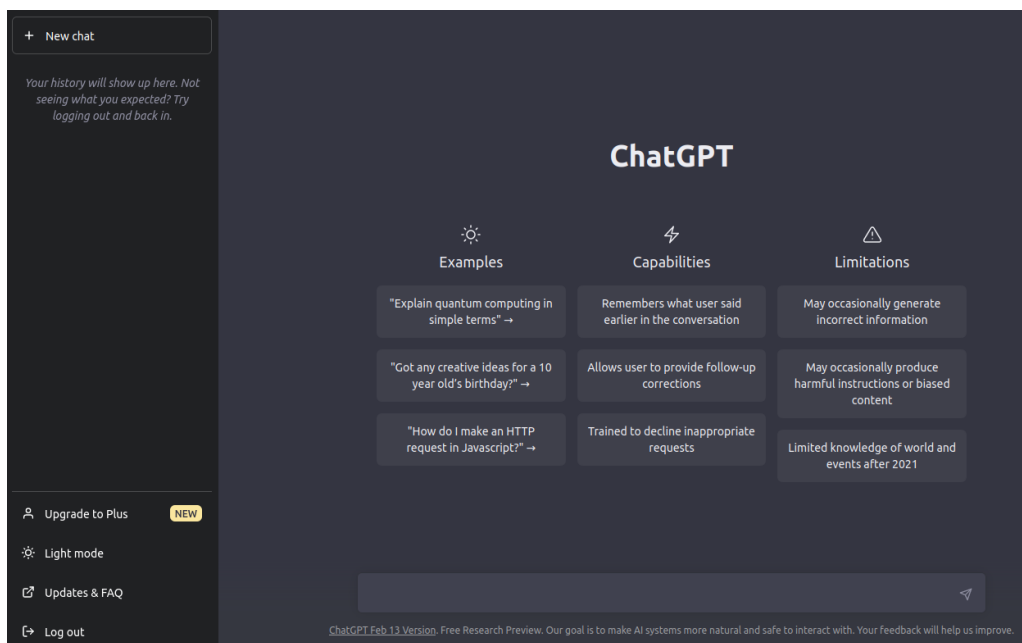


Figura 2-22 Interfaz de ChatGPT. [76]

Es un sistema *chat* diseñado para la asistencia a clientes y realización de búsquedas en diversos ámbitos. Uno de los mayores avances que ChatGPT ha implementado respecto a los *chatbots* tradicionales es la prácticamente idéntica similitud a lo que podría ser una conversación mantenida entre dos personas de modo natural y fluido, de tal manera que es capaz de obtener respuestas a cuestiones realizadas con lenguaje coloquial o formal en diferentes idiomas, así como realizar preguntas en caso de no haber comprendido bien aquello a lo que el usuario se refiere.

Pese a las provechosas y ventajosas virtudes que esta herramienta posee, su desarrollo ha traído consigo una serie de controversias no quedando exento de polémicas, pues muchos expertos aseguran que se ha convertido en una herramienta que fomenta la copia y el fraude en diferentes ámbitos: desde el alumnado de colegios y universidades hasta en proyectos empresariales. Otra de las críticas propiciadas por esta herramienta ha sido el libre acceso a códigos de proyectos a los que se les podría dar un uso cuyas consecuencias disten bastante de las deseadas. Es por ello que, OpenAI ha decidido desarrollar una API con la que poder filtrar y mantener monitorizados a los desarrolladores que pretendan acceder a dichos códigos. [77] Cabe destacar que muchas de las funcionalidades que esta API ofrece no son gratuitas como se mostrará en el desarrollo del proyecto. Para poder tener acceso a ella es necesario rellenar un cuestionario. No obstante, a pesar de ello, el crecimiento de esta herramienta conversacional de inteligencia artificial ha sido tal que, en apenas dos meses, alcanzó 100 millones de usuarios, lo que les supuso a redes sociales tales como Instagram, entre otros, más de dos años. En la Figura 2-23 se muestra una gráfica en la cual se analizan la cantidad de meses que le ha llevado a cada una de las aplicaciones llegar a la cifra de 100 millones de usuarios.

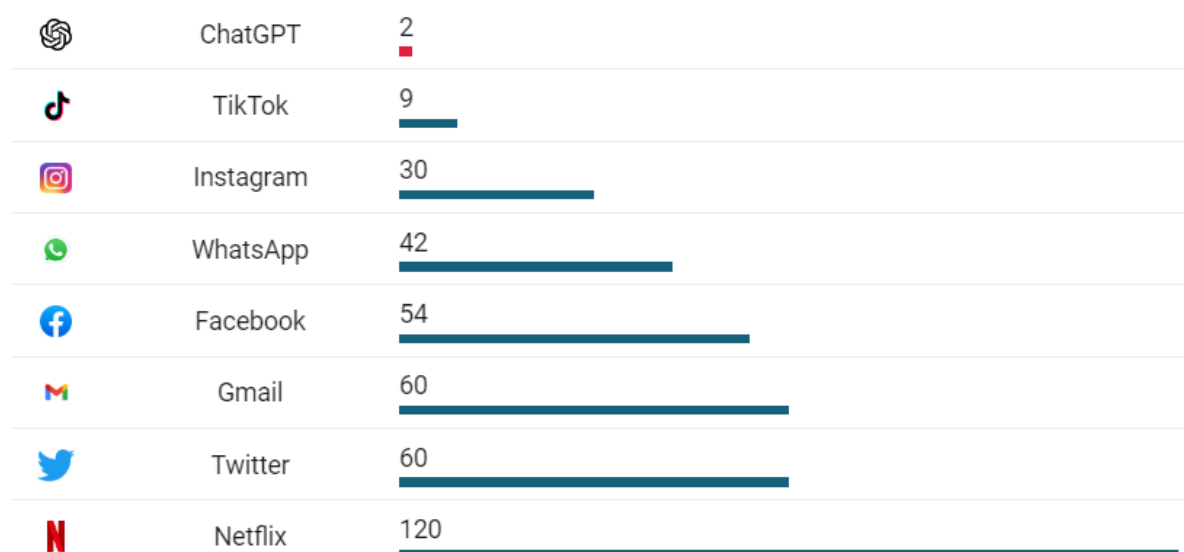


Figura 2-23 Meses transcurridos hasta llegar a 100 millones de usuarios. [80]

La versión más actualizada accesible sin restricciones en este momento es la versión ChatGPT-3.5, siendo la nueva generación del modelo de OpenAI: GPT-4 [81], accesible solo para usuarios con ChatGPT Plus (*premium*) con un coste de veinte dólares al mes. También se puede acceder mediante el buscador Bing de Microsoft pues implementa la última versión GPT-4. Esta última versión ofrece un mayor abanico de respuestas y evita que el *chatbot* se pierda en medio de una conversación, así como también aumenta la creatividad ofrecida al usuario pudiendo llegar a escribir guiones cinematográficos, componer canciones, etc. Esta versión, aseguran, podrá analizar y procesar imágenes de una manera exhaustiva para responder a preguntas que sobre ellas se le formulen. Aumenta, a su vez, la capacidad de obtener patrones de múltiples conversaciones, así como de las previas preguntas y contestaciones de una misma conversación y usuario lo que posibilitará una futura respuesta más precisa y válida. GPT-4 se ha implementado ya en asistentes de voz, así como en buscadores, como el ya mencionado: Bing. [82]

2.6 Trabajos académicos relacionados

2.6.1 Herramientas de OSINT para la obtención de información en Twitter sobre el conflicto Ucrania-Rusia

Trabajo Fin de Grado (TFG) realizado por Álvaro Hernández Ruiz (2021) en el Centro Universitario de la Defensa (en la Escuela Naval Militar) [83]. En este proyecto, se desarrolla un estudio exhaustivo de las diferentes herramientas de OSINT, de las cuales emplea T-hoarder. Mediante esta herramienta procede al análisis de los diferentes tuits publicados sobre el conflicto ruso-ucraniano. Obtiene a su vez, estadísticas sobre los diferentes usuarios que proporcionan información en la plataforma acerca de este conflicto, destacando aquellos que poseen mayor influencia en el ámbito, ya sea mediante el análisis de retuits, mayor número de tuits publicados, usuarios que posean más menciones, etc.

2.6.2 Detección de comunidades en redes: Algoritmos y aplicaciones

Realizado por Julio Omar Ancízar Palacio Niño (2013) [53]. Este proyecto fin de máster trata sobre el análisis de diferentes métodos de detección de comunidades, mediante el empleo de teorías de grafos entre otros, en redes sociales. Se hace un estudio sobre la complejidad que cada procedimiento presenta, así como las limitaciones de cada uno.

3 DESARROLLO DEL TFG

Para el desarrollo de la herramienta de detección de usuarios de interés para el ámbito de la defensa, se ha de seguir una determinada metodología compuesta por los pasos que se muestran en el esquema realizado en la Figura 3-1.

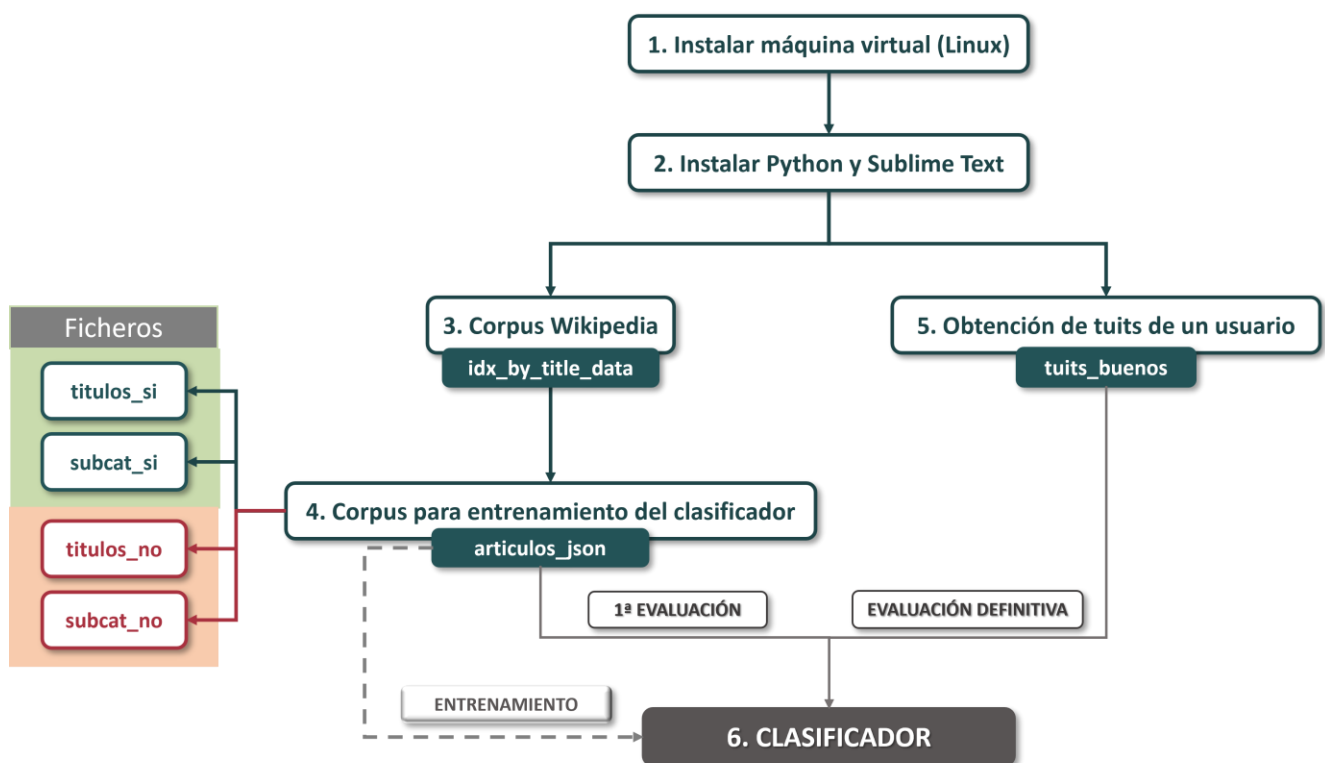


Figura 3-1 Metodología para la implementación del sistema de detección. (Autoría propia)

1. En primer lugar, se procederá con las indicaciones sobre la instalación del entorno con el que se llevará a cabo el proyecto.
2. Una vez instalada la máquina virtual se indicará cómo instalar a su vez el lenguaje de programación a emplear (*Python*), así como el editor de texto con el que se pretende trabajar (*Sublime Text*).
3. Se procederá entonces a la explicación detallada de la creación del corpus de Wikipedia del que se extraerán artículos tanto pertenecientes como no relacionados con dicho ámbito.

- Recurriendo al corpus de Wikipedia, se creará el archivo de texto *articulos_json*, el cual se empleará como base de datos para entrenar el clasificador.
- Se indica cómo extraer los cien últimos tuits de un usuario mediante la API de Twitter, y cómo proceder a realizar un preprocesado del texto obteniendo el volcado en el archivo de texto *tuits_buenos*.
- Finalmente, se procede a la generación e implementación del clasificador y a su entrenamiento. Las diferentes pruebas y evaluaciones del clasificador se detallarán en el siguiente capítulo.

3.1 Instalación de la máquina virtual

Para la realización del presente trabajo será necesario en primer lugar la creación de un entorno de trabajo adecuado. Con este fin, se ha decidido operar con el sistema operativo Linux, no siendo estrictamente necesario, pero sí recomendable para facilitar el desarrollo del sistema de detección inteligente. Para evitar realizar una partición del disco para la instalación de dicho sistema operativo, existe la posibilidad de instalar una máquina virtual. Para su creación habrá que acceder a la página oficial de VirtualBox [84] y proceder a la descarga e instalación de la máquina que se adecúe al sistema operativo del ordenador con el que se proceda a realizar el proyecto.

Una vez instalado el programa, se procederá a dar de alta la propia máquina virtual siguiendo los siguientes pasos:

- Hacer clic en “Nueva” para crear una nueva máquina virtual de Linux y, concretamente con la distribución de Ubuntu (véase Figura 3-2).



Figura 3-2 Nueva máquina virtual. (Autoría propia)

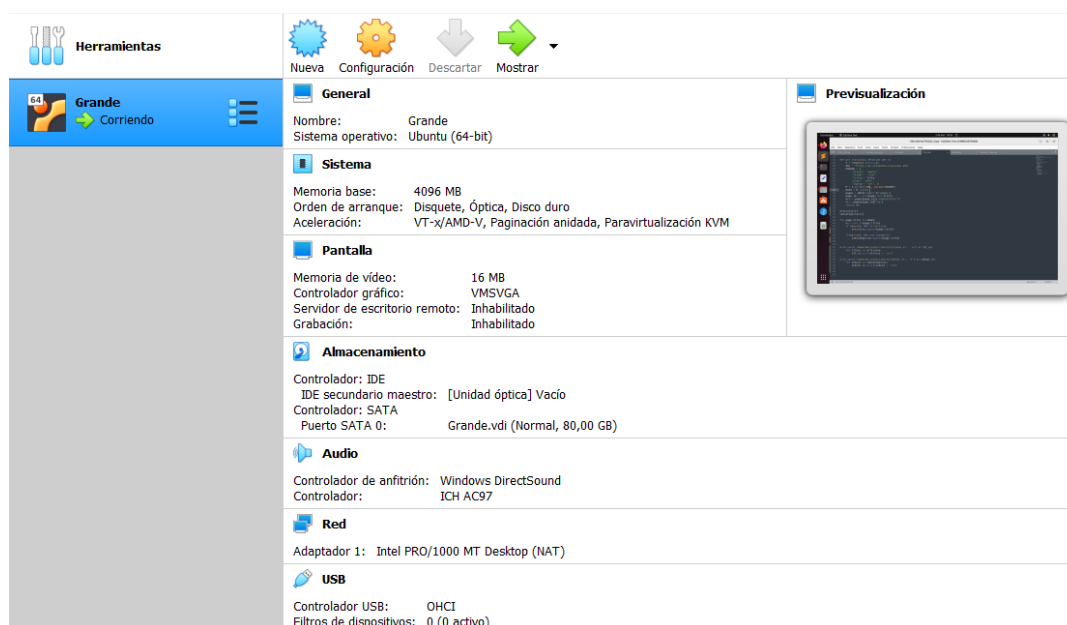


Figura 3-3 Configuración máquina virtual. (Autoría propia)

2. Será necesario configurar los siguientes parámetros: tamaño de la memoria RAM, espacio total de disco y el controlador gráfico mediante las ventanas que vayan apareciendo y según los intereses del proyecto en cuestión. En este trabajo se han establecido los parámetros que se pueden observar en la Figura 3-3.
3. Finalmente, aparecerá una interfaz a través de la cual se procede a instalar la máquina virtual tras haber seleccionado la correspondiente imagen ISO del sistema operativo.

3.2 Instalación de *Python* y *Sublime Text*

Python [85] es un lenguaje de programación de alto nivel, interpretado y cuya simplicidad, elegancia y versatilidad lo han convertido en uno de los lenguajes más empleados a día de hoy. Fue creado por Guido van Rossum a finales de los años ochenta. El indicador de popularidad *TIOBE Programming Community* [86] sitúa dicho lenguaje de programación en primera posición (véase Figura 3-4 y Tabla 3-1). Este índice se encuentra actualizado al presente mes de marzo del año 2023 y basa sus indicios en el número de ingenieros y personas con cualificación oficial a nivel global que emplean dicho lenguaje y en el análisis de motores de búsqueda como Google, Bing, YouTube, etc. Se trata de un lenguaje interpretado, lo que significa que no precisa su compilación para poder ejecutar los códigos, tan solo es necesario un programa intérprete.

Programming Language	2023	2018	2013	2008	2003
Python	1	4	8	7	13
C	2	2	1	2	2
Java	3	1	2	1	1
C++	4	3	4	4	3
C#	5	5	5	8	10
Visual Basic	6	15	-	-	-
JavaScript	7	7	11	9	8
Assembly language	8	12	-	-	-
SQL	9	251	-	-	7
PHP	10	8	6	5	6

Tabla 3-1 Ranking TIOBE para lenguajes de programación a lo largo de los años. [86]

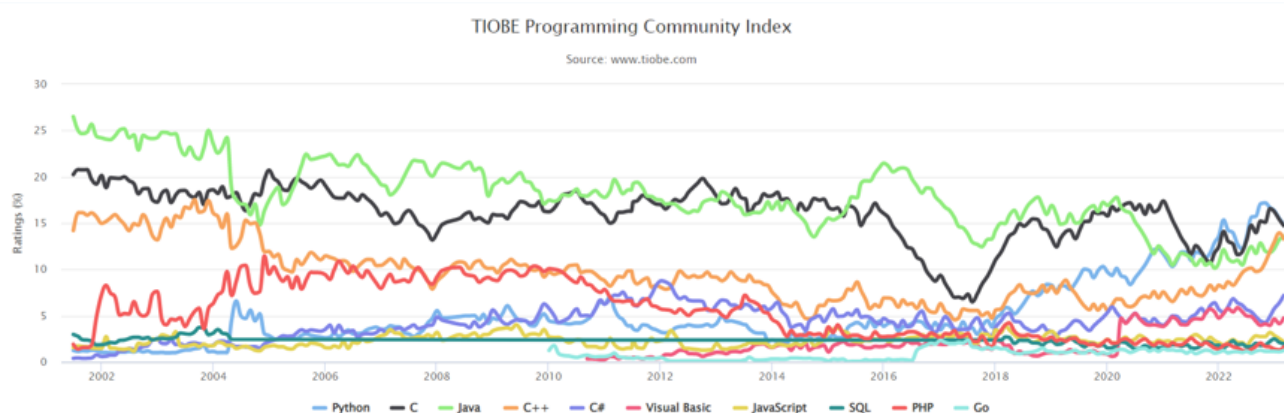


Figura 3-4 Índice TIOBE para lenguajes de programación. [86]

Una de las principales ventajas de este lenguaje es su sintaxis clara y concisa, lo que conlleva una fácil lectura y escritura. Cabe destacar la amplia variedad de bibliotecas y módulos que facilitan la programación en distintos ámbitos. Otra gran ventaja es que consiste en un lenguaje multiplataforma,

lo que implica que puede utilizarse en diferentes sistemas operativos como Windows, Linux o macOS. Se trata de un lenguaje de código abierto por lo que cualquier persona puede contribuir a su evolución.

Previo al desarrollo del sistema de detección inteligente, es necesaria la instalación de dicho lenguaje de programación en su versión 3.10.10 de alto nivel desde el terminal de la máquina virtual mediante el comando que se muestra en la Figura 3-5:

```
maria@maria-VirtualBox:~$ sudo apt install python3
```

Figura 3-5 Comando para instalar *Python*. (Autoría propia)

Será preciso, a su vez, la instalación de un editor de texto. En este caso se ha seleccionado *Sublime Text* [87]. Dicho editor fue lanzado en el año 2008 y se ha convertido en uno de los más populares y empleados por los desarrolladores de código. Cabe destacar la variedad de lenguajes con los que se puede trabajar mediante este editor, así como el autocompletado de texto y la capacidad de destacar la sintaxis mediante el empleo de colores que facilitan su visualización. Para llevar a cabo su instalación se empleará el comando de la Figura 3-6.

```
maria@maria-VirtualBox:~$ sudo apt install sublimetext
```

Figura 3-6 Comando para instalar *Sublime Text*. (Autoría propia)

3.3 Corpus de Wikipedia

Para comenzar, será necesario cargar el conjunto de datos de Wikipedia, mediante un código que permita su volcado. En este caso se empleará la versión de Wikipedia del 1 de marzo de 2022 en inglés como se muestra más adelante. Este volcado de todo el conjunto de artículos será imprescindible para poder extraer un conjunto que pertenezcan al ámbito de la defensa y un conjunto que no y, poder obtener así un corpus con el que poder entrenar al clasificador. Cabe destacar que dichos artículos se encuentran sanitizados para suprimir referencias, vínculos, imágenes, etc.

Se tomará como referencia la página oficial de *Hugging Face* [88], empresa tecnológica enfocada al desarrollo de herramientas de procesamiento del lenguaje natural en la cual se establecen unas directrices en cuanto a herramientas e instalaciones de librerías se refiere.

En primer lugar, los artículos se analizarán empleando la herramienta: *mwparserfromhell* [88], y mediante *Apache Beam* se procederá a la carga de los mismos. Para ello, se procederá a escribir el siguiente comando, extraído de la página de *Hugging Face* [88], en el terminal (véase Figura 3-7):

```
maria@maria-VirtualBox:~$ pip3 install apache_beam mwparserfromhell
```

Figura 3-7 Comando para instalar *Apache Beam* y *mwparserfromhell*. (Autoría propia)

Habrá que instalar, a su vez, en caso de no tenerla ya instalada, la librería *Hugging Face Datasets* [89] (desarrollada y mantenida por la misma empresa), la cual permitirá la descarga, procesamiento y empleo de los datos de Wikipedia. Habrá que escribir el comando de la Figura 3-8 en el terminal y así poder emplearla en el proyecto.

```
maria@maria-VirtualBox:~$ pip3 install datasets
```

Figura 3-8 Comando para instalar la biblioteca *datasets*. (Autoría propia).

Al finalizar estos pasos, se habrá descargado un *dump* (registro del contenido en un momento concreto) de Wikipedia de 20GB. Cabe la posibilidad de acceder a los datos del *dump* mediante el código mostrado en la Figura 3-9, lo cual creará un índice en un diccionario de *Python* que facilitará la recuperación de artículos mediante sus respectivos títulos. Este código de *Python*, mediante la función '*datasets.load_dataset()*' de la biblioteca *Hugging Face Datasets*, carga el conjunto de datos de Wikipedia en inglés correspondientes a la actualización del 1 de marzo de 2022. La función toma dos argumentos: el nombre del conjunto de datos (*Wikipedia*) y la versión a cargar (*20220301.en*). Luego,

el código itera sobre los artículos y crea el diccionario *idx_by_title* que asigna a cada título una posición en su índice. Finalmente, imprime en pantalla el índice junto con el título del artículo asociado.

```
1  import datasets
2
3  wikipedia = datasets.load_dataset("wikipedia", "20220301.en")
4
5  idx_by_title = {}
6  idx = 0
7  for article in wikipedia['train']:
8      title = article['title']
9      idx_by_title[title] = idx
10     print(idx, " ", title)
11     idx = idx + 1
```

Figura 3-9 Código de Python para la creación del índice *idx_by_title*. (Autoría propia)

Para acceder al artículo a partir de su título habrá que ejecutar el siguiente código de Python (véase Figura 3-10). En la primera línea de código se obtendría la posición que ocupa el artículo ‘Madrid’ en el diccionario creado. En la segunda línea de código habría que introducir la posición obtenida (en este caso la 452648) para acceder al título del artículo.

```
13  print (idx_by_title['Madrid'])
14  print (wikipedia['train'][452648])
```

Figura 3-10 Código de Python para obtener un artículo de Wikipedia mediante el título. (Autoría propia)

Se imprimiría en pantalla el artículo en formato de texto JSON (*Java Script Object Notation*) [90]. JSON utiliza una sintaxis sencilla para representar los datos y destaca por ser fácilmente procesable por humanos y máquinas. Como se puede ver reflejado en Figura 3-11 aparece el identificador interno de la página de Wikipedia, conocido como *page_id* o *id*; la URL (*Uniform Resource Locator*, para referirse a la dirección web única que identifica cada página web de Internet) del artículo, el título y el texto de este.

```
{'id': '1',
 'url': 'https://simple.wikipedia.org/wiki/April',
 'title': 'April',
 'text': 'April is the fourth month...'
}
```

Figura 3-11 Formato JSON de los artículos en el corpus de Wikipedia. [88]

Se procede a guardar el diccionario (véase código de la Figura 3-12) en un archivo mediante el módulo *pickle* [91], el cual se utiliza para serializar y deserializar una estructura de objetos de Python.

```

1  import datasets
2  import pickle
3
4  wikipedia = datasets.load_dataset("wikipedia", "20220301.en")
5
6  idx_by_title = {}
7  idx = 0
8  for article in wikipedia['train']:
9      title = article['title']
10     idx_by_title[title] = idx
11     print(idx, " ", title)
12     idx = idx + 1
13
14  with open('idx_by_title_data.pkl', 'wb') as diccionario:
15      pickle.dump(idx_by_title, diccionario)
16      print('Diccionario creado')

```

Figura 3-12 Código de *Python* para guardar el diccionario en un archivo. (Autoría propia)

Se guarda el diccionario de Wikipedia en el archivo denominado *idx_by_title_data.pkl*. Abre el archivo en modo binario para escritura ‘wb’ y la función ‘*pickle.dump()*’ es la encargada del volcado del diccionario en el archivo. Finalmente, imprime el mensaje “Diccionario creado” para verificar la creación y el correcto guardado.

Se podrá recurrir a este diccionario en otros códigos de *Python* mediante la implementación de las siguientes líneas (véase Figura 3-13):

```

1  import datasets
2  import pickle
3
4  wikipedia = datasets.load_dataset("wikipedia", "20220301.en")
5
6  with open('/home/maria/idx_by_title_data.pkl', 'rb') as diccionario:
7      idx_by_title=pickle.load(diccionario)

```

Figura 3-13 Código de *Python* para abrir el diccionario. (Autoría propia)

3.4 Generación de corpus para entrenamiento del clasificador

3.4.1 API de Wikipedia

La API de Wikipedia [92] es una interfaz de programación de aplicaciones proporcionada por la Fundación Wiki Media que permite el acceso a diferentes datos y contenidos de Wiki Media mediante diferentes métodos de programación con distintos lenguajes. Permite a los desarrolladores la consecución de información y contenido de una página, así como imágenes y otros metadatos de manera estructurada pertenecientes a este sitio web.

Dicha API utiliza el protocolo HTTP y devuelve los datos en formato JSON o XML. Es empleada por diversos desarrolladores con el objetivo de crear motores de búsqueda, *chatbots*, herramientas de minería de datos, etc.

Una de las funcionalidades que ofrece esta interfaz [92], y que más importancia va a tener en el desarrollo de este trabajo, es la posibilidad de obtener información acerca de categorías, subcategorías y artículos de Wikipedia, así como la relación existente entre ellas. Cabe destacar que el acceso a esta interfaz es totalmente gratuito y de código abierto, lo que supone que los desarrolladores pueden acceder y contribuir libremente al código fuente. Todo esto conlleva la posible aplicación de esta interfaz de programación en diversos ámbitos y proyectos de investigación y desarrollo. Si bien es cierto que tiene ciertas limitaciones. Por ejemplo, limita el número de solicitudes que se pueden hacer a la API durante un tiempo determinado. Otro inconveniente, es la fiabilidad de los artículos colgados en Wikipedia, puesto que al igual que el código, pueden ser editados por cualquier persona. Esto conlleva información inexacta e incompleta en ciertos casos.

3.4.2 Selección de categorías

Con el objetivo de poder discernir entre tuits, y por tanto usuarios, que pertenezcan al ámbito de la defensa de aquellos que no, se ha de generar un clasificador y entrenarlo con un corpus. Por tanto, será necesario desarrollar dicho corpus para que pueda funcionar como entrenamiento en el clasificador. Es por ello por lo que se han de extraer una serie de artículos con los que poder formar este conjunto de datos.

Se han de seleccionar una serie de categorías, tanto pertenecientes a la temática de defensa como no, para poder generar dicho corpus. Será preciso el establecimiento de unas categorías principales sobre las que partir la búsqueda de artículos y subcategorías que de esta deriven como se puede observar en la estructura ejemplo del proceso que a continuación se adjunta (véase Figura 3-14):

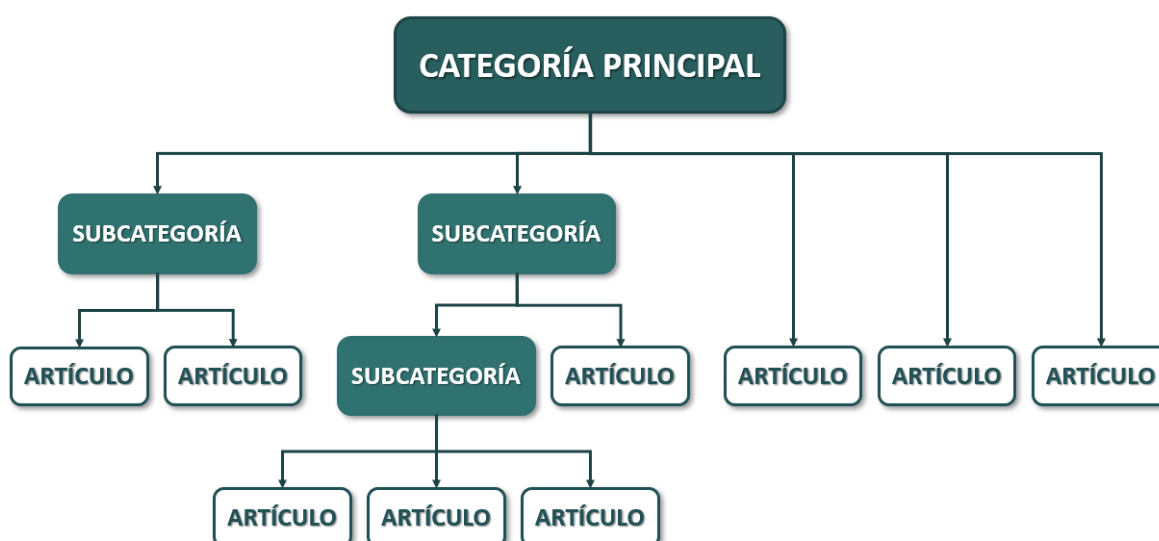


Figura 3-14 Estructura base para la obtención de artículos. (Autoría propia)

En cuanto a la obtención las categorías principales, se han empleado los estándares técnicos globales y eficientes de medios comunicación, definidos por el Consejo Internacional de Prensa de Telecomunicaciones (IPTC, *International Press Communications Council*) [93]. Estos estándares engloban desde metadatos de imágenes, vídeos, temas principales de los medios de noticias, etc. El objetivo principal de estos es funcionar como intermediarios entre consumidores y creadores de contenido y así mejorar su intercambio de información. En este caso, se emplearán las categorías principales, estipuladas por dicho Consejo, en las que se podrían clasificar todas las noticias (véase Tabla 3-2):

1	Arte, cultura y espectáculos
2	Policía y justicia
3	Catástrofes y accidentes
4	Economía negocios y finanzas
5	Educación
6	Medio ambiente
7	Salud
8	Interés humano, animales, insólito
9	Mano de obra
10	Estilo de vida y tiempo libre
11	Política

12	Religión y cultos
13	Ciencia y tecnología
14	Sociedad
15	Deporte
16	Conflicto, guerra y paz
17	Meteorología

Tabla 3-2 Categorías principales establecidas por el IPTC. (Autoría propia)

En la Tabla 3-2 se señalan en color azul aquellas categorías que tienen menor relación directa con el ámbito de la defensa y en rojo aquellas que sí están directamente relacionadas. Una vez listadas las categorías, es necesario asociarlas con categorías de la Wikipedia en inglés [94] (véase ejemplo de la categoría “*National security*” en la Figura 3-15 en la que se aprecian tanto las subcategorías como los artículos derivados de la propia categoría).

WIKIPEDIA

The Free Encyclopedia

Search Wikipedia

Create account

Log in

...

Category:National security

79 languages

CategoryTalk

ReadEditView history

From Wikipedia, the free encyclopedia

?

Help

The main article for this category is *National security*.

National security refers to policy enacted by governments to ensure the survival and safety of the nation-state, including but not limited to the exercise of diplomatic, economic, and military power in both peace and war.

Wikimedia Commons has media related to *National security*.

Contents

Top · 0-9 · ABCDEFGHIJKLMNOPQRSTUVWXYZ

Subcategories

This category has the following 30 subcategories, out of 30 total.

▶ National security by country (21 C, 6 P)

B

▶ Border guards (14 C, 81 P)

C

▶ Classified information (4 C, 27 P)

▶ Continuity of government (3 C, 23 P)

▶ Counterterrorism (14 C, 130 P)

▶ Curfews (3 C, 8 P)

D

▶ Data security (7 C, 137 P)

▶ Diplomacy (26 C, 150 P)

E

▶ Emergency laws (10 C, 51 P)

▶ Energy security (20 P)

▶ Security engineering (9 C, 41 P)

▶ Espionage (26 C, 89 P)

G

▶ Security sector governance and reform (7 P)

I

▶ Information sensitivity (6 C, 42 P)

▶ National security institutions (9 C, 48 P)

▶ Intelligence agencies (21 C, 43 P)

L

▶ Law enforcement (55 C, 152 P)

M

▶ Maritime security of India (1 C, 6 P)

▶ Military (42 C, 14 P)

P

▶ Government paramilitary forces (17 C, 46 P)

▶ Passports (8 C, 55 P, 1 F)

▶ National security policies (4 C, 16 P)

▶ Political repression (16 C, 82 P)

▶ Privacy of telecommunications (5 C, 92 P)

▶ Public safety (15 C, 37 P)

S

▶ Security studies (4 C, 62 P)

▶ Surveillance (24 C, 237 P)

T

▶ Terrorism (18 C, 24 P)

W

▶ Warning systems (8 C, 92 P)

▶ WikiLeaks (6 C, 46 P)

Pages in category "National security"

The following 95 pages are in this category, out of 95 total. This list may not reflect recent changes.

▪ National security

▪ National Security Advisor to the Vice President

*

▪ Security agency

A

▪ Advance Passenger Information System

B

▪ BuggedPlanet

C

▪ Climate change adaptation

▪ Climate security

G

▪ Ghost detainee

▪ Global surveillance

▪ List of government mass surveillance projects

▪ Greek wiretapping case 2004–05

H

▪ High policing

▪ Homeland security

▪ Hyper-surveillance

I

▪ Information security

▪ Information Security Automation Program

▪ Information sensitivity

▪ Infrastructure security

P

▪ Peace through strength

▪ Phone surveillance

▪ PhotoDNA

▪ Physical security

▪ Police

▪ Postal censorship

▪ Proactive cyber defence

Q

▪ Qatar Counter-terrorism Strategy

R

▪ Read into

▪ Reiss Center on Law and Security

▪ Rule by decree

Figura 3-15 Categoría de Wikipedia “*National security*” (Seguridad nacional). [95]

3.4.3 Obtención de artículos para la generación del corpus

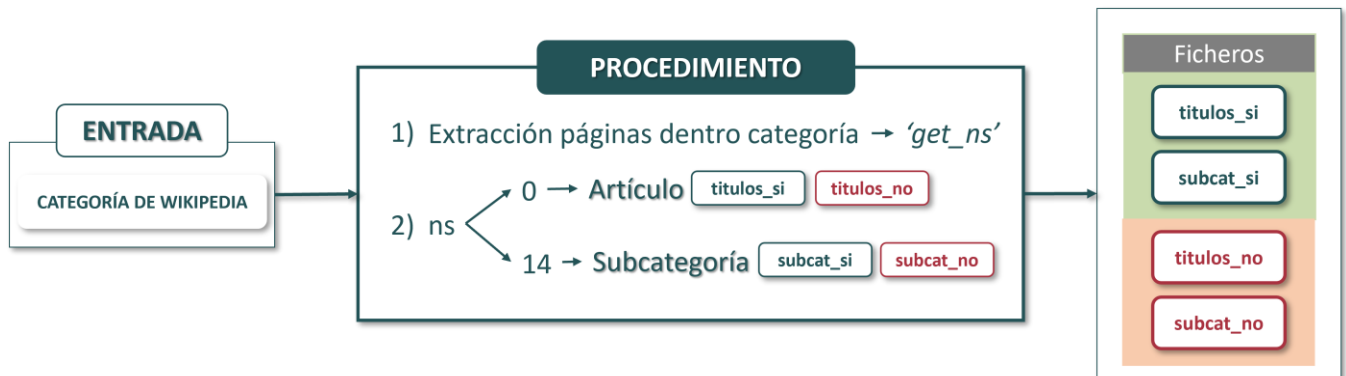


Figura 3-16 Proceso de obtención de artículos a partir de una categoría. (Autoría propia)

En esta primera Figura 3-16 se puede ver de manera esquemática el script creado para la obtención de los artículos en base a la categoría de Wikipedia seleccionada, como se explica a continuación.

Una vez fijadas las principales categorías se podrá comenzar a aplicar el código explicado en las siguientes líneas (véase Figura 3-17). En primer lugar, es necesario crear cuatro ficheros de texto. El primero de ellos contendrá títulos de artículos que sí estén relacionados con la temática de defensa (*titulos_si*); el segundo con aquellos que no guarden relación con dicha temática (*titulos_no*). El tercero y el cuarto tendrán como objetivo la recopilación de las subcategorías que vayan derivando de las diferentes categorías (*subcat_si* y *subcat_no*). Para la formación de estos ficheros se aplica el código mostrado y explicado a continuación.

```

1  import requests
2
3  S = requests.Session()
4  URL = "https://en.wikipedia.org/w/api.php"
5
6  CATEGORY_NAME = input("Ingrese el nombre de la CATEGORÍA: ")
7
8  PARAMS = {
9      "action": "query",
10     "format": "json",
11     "list": "categorymembers",
12     "cmtitle": "Category:" + CATEGORY_NAME,
13     "cmlimit": "max" }
14
15  R = S.get(url=URL, params=PARAMS)
16  DATA = R.json()
17
18  #Crea una lista de títulos de artículos dentro de la categoría
19  PAGES = [page["title"] for page in DATA["query"]["categorymembers"]]
20
21  def get_ns(title): #Funcion get_ns
22      S = requests.Session()
23      URL = "https://en.wikipedia.org/w/api.php"
24      PARAMS = {
25          "action": "query",
26          "format": "json",
27          "titles": title,
28          "prop": "info",
29          "inprop": "url", }
30      R = S.get(url=URL, params=PARAMS)
31      DATA = R.json()
32      pages = DATA["query"]["pages"]
33      page_id = list(pages.keys())[0]
34      url = pages[page_id]["canonicalurl"]
35      ns = pages[page_id]["ns"]
36      return ns
37
38  articulos=[]
39  subcategorias=[]
40
41  for page_title in PAGES:
42      ns = get_ns(page_title)
43      if (ns==0): #Es un artículo
44          articulos.append(page_title)
45      if(ns==14): #Es una categoría
46          subcategorias.append(page_title)
47
48  with open('/home/maria/Escritorio/titulos_si', 'a') as tit_si:
49      for titulo in articulos:
50          tit_si.write(titulo + "\n")
51  with open('/home/maria/Escritorio/subcat_si', 'a') as subcat_si:
52      for subcat in subcategorias:
53          subcat_si.write(subcat + "\n")

```

Figura 3-17 Código de Python para obtener artículos y subcategorías pertenecientes a una categoría. (Autoría propia)

En la primera línea se puede observar como este código importa la biblioteca *requests* la cual permite conectarse con la API de Wikipedia. Para ello crea una sesión: '*requests.Session()*' donde se especifica la URL de la API de Wikipedia con la que se pretende establecer la conexión. Tras esto, se solicita el parámetro de entrada del nombre de la categoría a estudiar y que debe ser modificado por el desarrollador en cada iteración según se proceda con las diferentes categorías seleccionadas a medida que se quiera ir extrayendo artículos y subcategorías de estas.

Se procede a proporcionar los parámetros de la solicitud que se va a realizar a la API para obtener la información deseada (en este caso serán todos los artículos y subcategorías que pertenecen a esta categoría), siendo el formato de salida en JSON. La respuesta de la API se almacenará en la variable

‘*DATA*’. Luego, se define una función denominada ‘*get_ns ()*’ que recibe como único parámetro el título de una página de Wikipedia y utiliza la API para obtener su *namespace* (ns) [96], (identificador numérico que diferencia el tipo de página al que pertenece) pudiendo tomar todos los que aparecen en la Tabla 3-3. En base a esta, será necesario discernir entre categoría y artículo.

NAMESPACE (ns)			
Espacios básicos		Espacios de discusión	
0	Principal (o artículo)	Discusión	1
2	Usuario	Usuario discusión	3
4	Wikipedia	Wikipedia discusión	5
6	Archivo	Archivo discusión	7
8	MediaWiki	MediaWiki discusión	9
10	Plantilla	Plantilla discusión	11
12	Ayuda	Ayuda discusión	13
14	Categoría	Categoría discusión	15
100	Portal	Portal discusión	101
102	Wikiproyecto	Wikiproyecto discusión	103
104	Anexo	Anexo discusión	105
828	Módulo	Módulo discusión	829
2300	Accesorio	Accesorio discusión	2301
2302	Accesorio definición	Accesorio definición discusión	2303
Espacios virtuales			
-1	Especial		
-2	Medio		

Tabla 3-3 Namespace (ns) de las páginas de Wikipedia. (Autoría propia)

Esto es precisamente lo que hace la función ‘*get_ns ()*’ la cual devuelve el *ns* correspondiente al título de la página que le entra como argumento. Una vez definida la función se recorre el array formado por los títulos de las páginas ‘*PAGES*’, se emplea la función ‘*get_ns ()*’ de tal manera que si el *ns*=0, es decir, se trata de un artículo, imprime el título en el archivo *titulos_si*. Si, sin embargo, el *ns*=14 se trata de una subcategoría e imprimirá la subcategoría en un archivo de texto denominado *subcat_si*.

De este modo, se obtendrán dos archivos de texto: uno con todos los artículos que pertenecen a las categorías (véase ejemplo en la Figura 3-18) que se vayan indicando en el código anterior, y otro con las subcategorías (véase ejemplo en la Figura 3-19) que nacen de esta categoría. Cabe destacar, que para crear otros dos ficheros equivalentes con los artículos y subcategorías que no pertenecen al ámbito de la defensa, habrá que cambiar simplemente el nombre del fichero de texto donde imprimir la salida en el código, así como introducir como parámetros de entrada categorías que no pertenezcan a este ámbito.

```

Abrir  [icono]  titulos_si
~/Escritorio

1 Spanish frigate Almirante Juan de Borbón
2 Spanish frigate Álvaro de Bazán
3 Álvaro de Bazán-class frigate
4 Spanish frigate Blas de Lezo (F103)
5 Spanish frigate Cristóbal Colón
6 Duc du Maine (slave ship)
7 HMAS Flinders (FFG)
8 Frigate
9 ARM Guadalupe
10 Chinese frigate Hai'an
11 HMAS Hunter
12 Incheon-class frigate
13 ARM Montezuma
14 HMAS Tasman
15 Valour-class frigate
16 List of frigate classes by country
17 Spanish frigate Santa Maria
18 HMS Amfritrite (1804)
19 French frigate Hélène (1791)
20 SS Royal William
21 Baleares-class frigate
22 Spanish frigate El Gamo
23 Spanish frigate Esmeralda
24 F110-class frigate
25 HMS Florentina (1800)
26 Galera Victoria

```

Figura 3-18 Archivo de texto *titulos_si*. (Autoría propia)

```

Abrir  [icono]  *subcat_si
~/Escritorio

1 Category:Harrier Jump Jet
2 Category:BAE Systems Hawk
3 Category:Hawker Siddeley aircraft
4 Category:Lists of British Aerospace aircraft operators
5 Category:Accidents and incidents involving the Harrier Jump Jet
6 Category:Hawker Hurricane
7 Category:Lists of Hawker aircraft operators
8 Category:Frigates by country
9 Category:Frigates by navy
10 Category:Frigate classes
11 Category:Frigates by period
12 Category:Frigates sunk by aircraft
13 Category:Lists of frigates
14 Category:Steam frigates
15 Category:Frigate templates
16 Category:Frigates of Argentina
17 Category:Frigates of Australia
18 Category:Frigates of Austria-Hungary
19 Category:Frigates of Bangladesh
20 Category:Frigates of Belgium
21 Category:Frigates of Brazil
22 Category:Frigates of Bulgaria
23 Category:Frigates of Canada
24 Category:Frigates of Chile
25 Category:Frigates of the People's Republic of China
26 Category:Frigates of the Republic of China

```

Figura 3-19 Archivo de texto *subcat_si*. (Autoría propia)

El siguiente paso será analizar el archivo de subcategorías relacionadas (o no) con el ámbito de la defensa para extraer a su vez sus artículos y subcategorías. El proceso será similar al anterior con la diferencia que en este caso recibe como entrada los ficheros de subcategorías como se observa en la Figura 3-20.

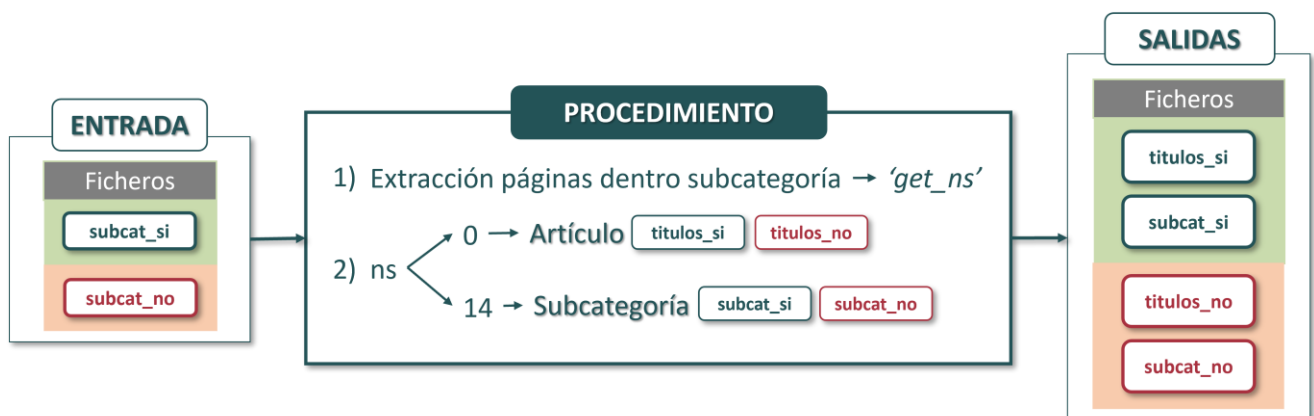


Figura 3-20 Proceso de obtención de artículos a partir de subcategorías (II). (Autoría propia)

Para ello se ha realizado el código de la Figura 3-21, mediante el cual se procesarán aquellas subcategorías recogidas en los archivos *subcat_si* y en *subcat_no*. De esta manera se agiliza y automatiza el proceso de seguir obteniendo títulos para ampliar el clasificador y hacerlo, de este modo, más preciso.

```

1  import requests
2
3  S = requests.Session()
4  URL = "https://en.wikipedia.org/w/api.php"
5
6  def get_ns(page_title): #Funcion get_ns
7      S = requests.Session()
8      URL = "https://en.wikipedia.org/w/api.php"
9      PARAMS = {
10         "action": "query",
11         "format": "json",
12         "titles": page_title,
13         "prop": "info",
14         "inprop": "url", }
15      R = S.get(url=URL, params=PARAMS)
16      DATA = R.json()
17      pages = DATA["query"]["pages"]
18      page_id = list(pages.keys())[0]
19      url = pages[page_id]["canonicalurl"]
20      ns = pages[page_id]["ns"]
21      return ns
22
23  articulos=[]
24  subcategorias=[]
25  nuevos = []
26  with open('/home/maria/Escritorio/abrir', 'r') as abrir:
27      nuevos = abrir.readlines()
28  nuevos = [line.replace("Category:", "").strip() for line in nuevos]
29
30  for cat_title in nuevos:
31      if cat_title.strip(): # Verifica que el título de la página no está vacío
32          CATEGORY_NAME = cat_title
33
34          PARAMS = {
35             "action": "query",
36             "format": "json",
37             "list": "categorymembers",
38             "cmtitle": "Category:" + CATEGORY_NAME,
39             "cmlimit": "max"
40          }
41          R = S.get(url=URL, params=PARAMS)
42          DATA = R.json()
43          PAGES = [page["title"] for page in DATA["query"]["categorymembers"]]
44          for page_title in PAGES:
45              ns = get_ns(page_title)
46
47              if (ns==0): #Es un artículo
48                  articulos.append(page_title)
49              if (ns==14): #Es una categoría
50                  subcategorias.append(page_title)
51
52  with open('/home/maria/Escritorio/titulos_no', 'a') as tit_si:
53      for titulo in articulos:
54          tit_si.write(titulo + "\n")
55  with open('/home/maria/Escritorio/subcat_no', 'a') as subcat_si:
56      for subcat in subcategorias:
57          subcat_si.write(subcat + "\n")

```

Figura 3-21 Código de *Python* para automatizar el procesamiento de subcategorías. (Autoría propia)

Este código recorre el archivo *subcat_si* (siendo *subcat_no* si se tratase del conjunto de subcategorías no relacionadas con defensa), y ejecuta el mismo proceso que en el código anterior (Figura 3-17) para cada una de las subcategorías contenidas en dicho fichero de texto. Cabe destacar que, al leer el archivo de las subcategorías, puesto que estas van precedidas por la cadena de texto 'Category:', se suprime con el fin de obtener el título de la subcategoría manera íntegra y así poder procesarlo.

Una vez se obtengan los dos archivos de texto con los títulos finales (*titulos_si*, *titulos_no*), se procederá a pasar un filtro para eliminar posibles repeticiones de títulos. Esto puede darse, pues cada

artículo puede pertenecer a más de una categoría o subcategoría. El código para el filtrado será el que se puede observar en la Figura 3-22.

```

1  with open('/home/maria/Escritorio/titulos_si', 'r') as abrir:
2      lineas = abrir.readlines()
3
4  titulos_sin_repetir = []
5  titulos_repetidos = []
6
7  for linea in lineas:
8      if linea not in titulos_sin_repetir:
9          titulos_sin_repetir.append(linea)
10         if linea in titulos_sin_repetir:
11             titulos_repetidos.append(linea)
12
13  with open('/home/maria/Escritorio/final_si', 'a') as final_si:
14      for linea in titulos_sin_repetir:
15          final_si.write(linea)
16
17  print(f"Se encontraron {len(titulos_repetidos)} títulos repetidos y se eliminaron del archivo.")

```

Figura 3-22 Código de Python para eliminar títulos repetidos. (Autoría propia)

Es entonces cuando se procede a volcar el JSON del conjunto completo de artículos, escogidos en los ficheros *titulos_si* y *titulos_no*, en un nuevo fichero denominado *articulos_json*. En el esquema de la Figura 3-23 se puede contemplar de manera sintetizada cuáles son las entradas y la salida del código mostrado a continuación (véase Figura 3-24).

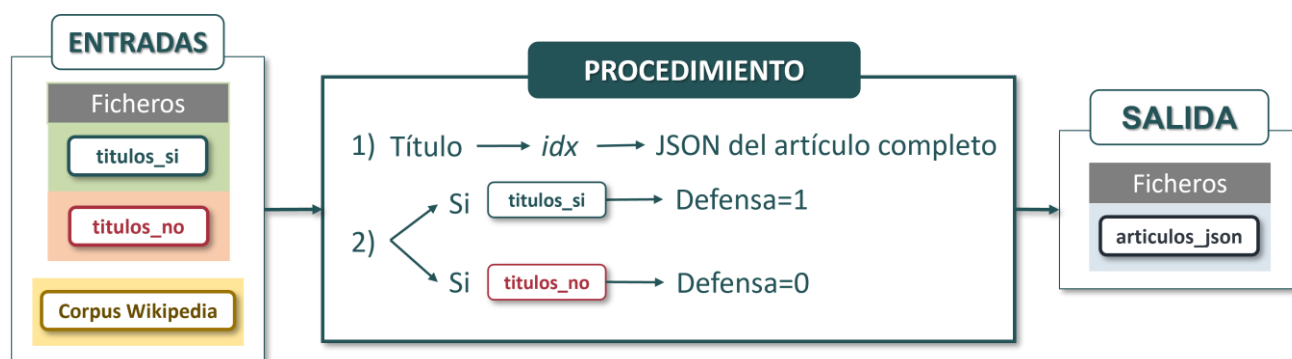


Figura 3-23 Proceso de obtención de los artículos en base a los títulos seleccionados. (Autoría propia)

Este código tiene como función principal el volcado del corpus creado para el posterior entrenamiento del clasificador en un fichero de texto con formato JSON denominado *articulos_json*. Para ello, en primer lugar, carga el corpus de Wikipedia utilizando la librería de *datasets* de *Hugging Face* e importa el índice, previamente creado y almacenado en un archivo, con ayuda de la biblioteca *pickle*. Cabe recordar que este índice contiene todos los títulos con identificadores específicos de los artículos existentes hasta la fecha 1 de marzo de 2022 de dicha enciclopedia.

En segundo lugar, lee las líneas de los ficheros, que contienen los títulos clasificados como pertenecientes y no al ámbito de la defensa, creados mediante los anteriores pasos: *titulos_si* y *titulos_no*. Para cada título se comprueba si existe en el diccionario cargado previamente. Si está presente, procede a obtener el índice correspondiente y recuperar la información (el texto) correspondiente a dicho artículo. Se agrega la etiqueta “Defensa” al diccionario JSON que vale 0/1 según el archivo de origen: *titulos_si* o *titulos_no*; es decir, si pertenece al ámbito de la defensa o no.

Los artículos clasificados positivamente se agregan a la lista denominada *articulos_si*, y los clasificados negativamente a la lista *articulos_no*. Se imprime en pantalla el número de artículos existentes en cada una de las listas. Finalmente, se escriben los artículos de estas dos últimas listas en formato JSON empleando la función ‘*writelines()*’ en modo *append*, de tal manera que permita ir agregando los artículos sin sobrescribir los anteriores.

```

1  import datasets
2  import pickle
3
4  wikipedia = datasets.load_dataset("wikipedia", "20220301.en")
5
6  with open('/home/maria/idx_by_title_data.pkl', 'rb') as diccionario:
7      idx_by_title=pickle.load(diccionario)
8
9  titulos_si=[]
10 titulos_no=[]
11 articulos_si=[]
12 articulos_no=[]
13
14 with open('/home/maria/Escritorio/titulos_si', 'r') as clas_si:
15     titulos_si=clas_si.readlines()
16
17     for titulo in titulos_si:
18         if titulo.strip() not in idx_by_title:
19             continue
20         idx=idx_by_title[titulo.strip()]
21         json_si=wikipedia['train'][idx]
22         json_si['Defensa']=1
23         articulos_si.append(json_si)
24
25 print("Número de artículos clasificados positivamente:", len(articulos_si))
26
27 with open('/home/maria/Escritorio/articulos_json', 'a') as fichero_si:
28     fichero_si.writelines("%s\n" % str(json_si) for json_si in articulos_si)
29
30
31 with open('/home/maria/Escritorio/titulos_no', 'r') as clas_no:
32     titulos_no=clas_no.readlines()
33
34     for titulo in titulos_no:
35         if titulo.strip() not in idx_by_title:
36             continue
37         idx=idx_by_title[titulo.strip()]
38         json_no=wikipedia['train'][idx]
39         json_no['Defensa']=0
40         articulos_no.append(json_no)
41
42 print("Número de artículos clasificados negativamente:", len(articulos_no))
43
44 with open('/home/maria/Escritorio/articulos_json', 'a') as fichero_no:
45     fichero_no.writelines("%s\n" % str(json_no) for json_no in articulos_no)

```

Figura 3-24 Código de Python para volcar el corpus en el fichero *articulos_json*. (Autoría propia)

El archivo resultado de ejecutar este código muestra la información en formato JSON, como en la Figura 3-25 se puede comprobar, con los siguientes parámetros: id de la página, la URL correspondiente, el título y el fragmento de texto extraído de dicho artículo.

```

1 {'id': '9288900', 'url': 'https://en.wikipedia.org/wiki/Spanish%20frigate%20Almirante%20Juan%20de%20Borb%C3%B3n', 'title': 'Spanish frigate
Almirante Juan de Borbón', 'text': "Almirante Juan de Borbón (F-102) is the second ship of the new F-100 class of air defence frigates entering
service with the Spanish Navy in 2003. She is named for Infante Juan de Borbón, Count of Barcelona, father of the former King of Spain, Juan
Carlos I.\n\nHistory\n September 8, 2006 - Departs from Rota to escort the Spanish troops en route to Lebanon\n November 12, 2006 - Arrives at
Rota, returning from Lebanon without incident\n February 2007 - Participates in GALIBER 07 Exercises\n March 2007 - Returns to port\n May 2007 -
Sails for trials at Cartagena\n June 2007 - Sails to Norway (Stavanger) and Greece (Chania and Heraklion) returning to base by the end of the
month\n January 2008 - Sails to Cádiz and is forced back by malfunction, expending some time in repairs.\n April 2008 - Sails to Brest (France)
to perform combat qualification with the French Navy\n June 2008 - Sails to Estonia and returns in December of the same year\n January 2010 -
Sails to Portugal and Cádiz\n May-June 2010 - Sails to perform combat qualifications\n January 12, 2011 - Arrives at Naval Station Norfolk,
Virginia, prior to the warship's participation with Carrier Strike Group Two during its Composite Unit Training Exercise (COMPTUEX) and Joint
Task Force Exercise (JTFEX) underway period.\n\nSpecifications\n Builder: Izo, Astillero Ferrol\n Propulsion: 2 x General Electric LM2500 gas
turbines, 2 x Caterpillar 3600 diesel engines\n Shafts: 2 (Wärtsilä controllable pitch propellers)\n Length: 146.7\m (482\ft)\n Beam:
18.6\m (61\ft)\n Draught: 4.75 (15\ft)\n Displacement: 5,800 tons (full load)\n Speed: 28.5 knots (52+ km/h)\n Range: 4500\m\n Cost:
600 million €\n Crew: 250 (48 officers)\n Armament:\n 1 x 5-inch/54 Mk45 Mod 2 gun\n 2 x CIWS FABA 20mm/120 Meroka gun\n 6 x Mk41 8-cell VLS\n
SAM: 32 x Standard SM-2 Block IIIA\n SAM: 64 x RIM-162 Evolved Sea Sparrow Missile\n SSM: 8 x McDonnell Douglas RGM-84 Harpoon anti-ship
missile\n ASW: 4 x 324\mm Mk32 Mod 9 triple Torpedo launchers with 12 Honeywell Mk46 mod 5 Torpedo\n Helicopter:\n 1 x Sikorsky SH-60B LAMPS
III Seahawk\n Electronics\n Sonar: ENOSA-Raytheon DE 1160LF (I)\n Radar: Lockheed Martin AN/SPY-1D 3-D multifunction\n Surface Radar: Raytheon
SPS-67(V)4\n Weapon Control: Aegis combat system, 2 x Raytheon SPG-62 Mk99 radar illuminator, 1 x FABA DORNA fire control\n Navigation: Thales
Scout\n Countermeasures\n 4 x FMC SRBOC Mk36 flare launchers\n SLQ-25A Enhanced Nixie torpedo countermeasures\n ESM/ECM: Indra SLQ-380\n CESELSA
Elnath Mk 9000 interceptor\n\nNotes\n\nExternal links\n\nFrigates\nFrigates of the Spanish Navy\nShips built in Spain\n2002 ships", 'Defensa': 1}
2 {'id': '1342702', 'url': 'https://en.wikipedia.org/wiki/Spanish%20frigate%20C3%81lvaro%20de%20Baz%C3%A1n', 'title': 'Spanish frigate Álvaro de
Bazán', 'text': "Álvaro de Bazán is the lead ship of the of air defence frigates entering service with the Spanish Navy. She is named after
Admiral Álvaro de Bazán.\n\nOperational history\nIn late 2005, Álvaro de Bazán was deployed as part of the aircraft carrier battle group in the
Persian Gulf. This was the first deployment of a Spanish warship as part of an American naval battle group.\n\nIn early March 2007, Álvaro de
Bazán became the first Spanish warship to visit Australia in 150 years. The deployment included several port visits, and was performed to support
Navantia's bid to design the for the Royal Australian Navy. The ship was also visiting Australia as part of the first circumnavigation of the
globe by a Spanish warship in 142 years.\n\nReferences\n\nExternal links\nBUQUESDEGUERRA.TK, a Spanish website about warships (Spanish)
\n\nFrigates\nFrigates of the Spanish Navy\nShips built in Spain\n2002 ships", 'Defensa': 1}

```

Figura 3-25 Fichero *articulos_json*. (Autoría propia)

3.5 Obtención de tuits de un usuario

3.5.1 API de Twitter

Para poder acceder a los datos de dicha red social, será necesaria la consecución de las credenciales pertinentes mediante la creación y solicitud de una cuenta de Twitter. Con este fin, habrá que acceder primero a la página oficial de Twitter, seleccionar crear una nueva cuenta y rellenar los datos que aparecen en el formulario mostrado en la Figura 3-26.

× Paso 1 de 5

Crea tu cuenta

Nombre

NOMBRE DE USUARIO

Teléfono

TELÉFONO

[Usar correo](#)

Fecha de nacimiento

Esta información no será pública. Confirma tu propia edad, incluso si esta cuenta es para una empresa, una mascota u otra cosa.

Mes

Día

Año

FECHA DE NACIMIENTO

Siguiente

Figura 3-26 Crear una cuenta de usuario en Twitter. (Autoría propia)

Tras completar los datos mostrados, se le enviará al usuario un código de confirmación a su teléfono móvil que tendrá que introducir en la nueva pestaña para su confirmación. El último paso a realizar será la selección de una contraseña y el idioma con el que se pretende hacer uso de la aplicación. Finalmente, se habrá registrado correctamente en la red social Twitter.

Una vez creada la cuenta de usuario, se procede a realizar la solicitud como desarrollador. Primero se accederá a la página oficial de la API de Twitter [33] y seleccionar en la parte superior de la derecha “Portal para desarrolladores” (véase Figura 3-27), donde se tendrá que iniciar sesión con la cuenta anteriormente creada.

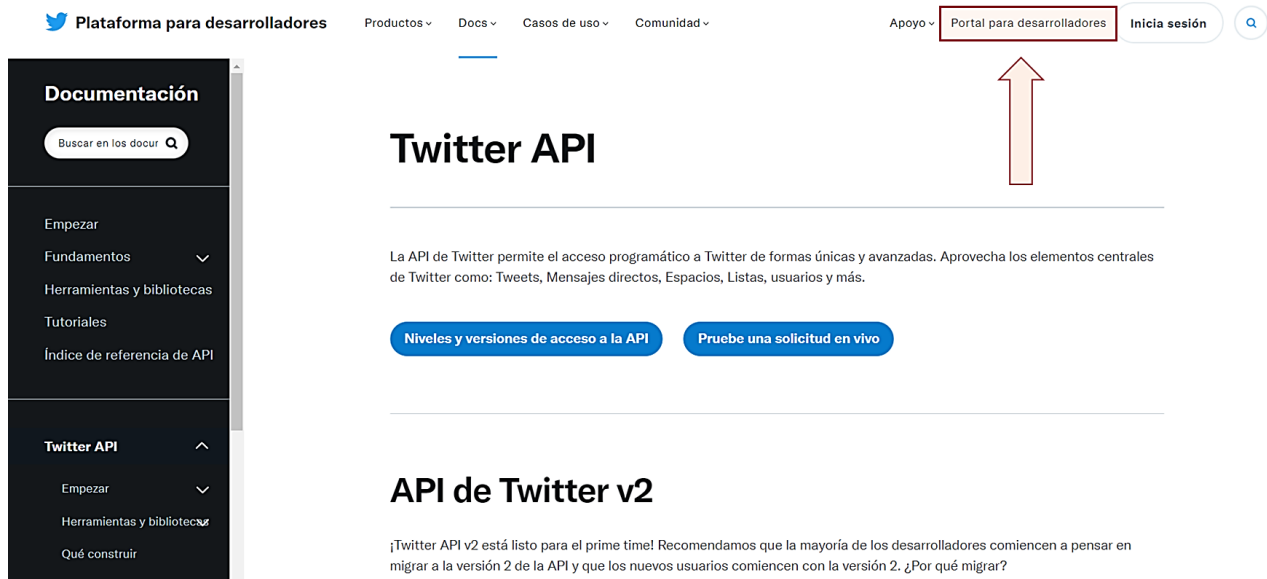


Figura 3-27 Twitter API. (Autoría propia)

Una vez se ha accedido al portal de desarrolladores con el usuario indicado, se procederá a la solicitud de las credenciales mediante la creación de un nuevo proyecto o aplicación. En caso, se seleccionará el nivel de credenciales para estudiante (*Student*). Habrá que especificar el país de pertenencia, así como el nombre en el desplegable, y finalmente el nombre que se pretenda otorgar al nuevo proyecto (véase secuencia de pasos en la Figura 3-28). Una vez aceptada la política de privacidad procederán a enviar otro código de verificación al número de teléfono especificado, y entonces la solicitud habrá sido elevada.

Figura 3-28 Pasos para crear un proyecto en el portal de desarrolladores. (Autoría propia)

Es importante conocer las limitaciones intrínsecas a cada nivel de acceso otorgado a cada desarrollador. Estas se pueden contemplar en la Tabla 3-4.

	Acceso	Nº Apps	Límite de Tuits (mensual)	Métodos de autenticación
Acceso esencial	Cualquier persona con cuenta de desarrollador	1	500.000	OAuth 2.0 con PKCE
				Solo aplicación
Acceso elevado	Mediante solicitud de acceso adicional	3	2.000.000	OAuth 2.0 con PKCE
				OAuth 1.0a
				Solo aplicación
Acceso a la investigación académica	Investigadores que cumplan una serie de requisitos	1	10.000.000	OAuth 2.0 con PKCE
				OAuth 1.0a
				Solo aplicación

Tabla 3-4 Limitaciones de los niveles de acceso para desarrolladores. (Autoría propia)

Con el proyecto creado y autorizado el acceso, se habrán otorgado al usuario una serie de claves personales (véase Figura 3-29) que serán necesarias (especialmente el *Bearer Token*) para autenticarse y poder realizar los códigos que permitan la obtención de la información de Twitter que será necesaria para la implementación de la herramienta a desarrollar.

The screenshot displays the 'Authentication Tokens' section of the Twitter Developer Portal. It includes the following elements:

- API Key and Secret:** A section with a 'Reveal API Key hint' link and a 'Regenerate' button.
- Authentication Tokens:** A section containing:
 - Bearer Token:** Generated on March 3, 2023, with 'Revoke' and 'Regenerate' buttons.
 - Access Token and Secret:** Generated on February 6, 2023, for the user '@mariafortrejon', with 'Revoke' and 'Regenerate' buttons. It also notes 'Created with Read and Write permissions'.
- OAuth 2.0 Client ID and Client Secret:** A section showing the 'Client ID' (b2dtQXZUJ...) and a 'Client Secret' (partially obscured by a green box), with a 'Reveal Client Secret hint' link and a 'Regenerate' button.

Figura 3-29 Claves de autenticación para desarrolladores. (Autoría propia)

3.5.2 Librería Tweepy

Para poder realizar los siguientes códigos en los que se implica a la red social Twitter será necesaria la descarga de la librería de *Python Tweepy* [97]. Esta permite trabajar e interactuar con la API de dicha red social (véase Figura 3-30), manteniendo las actualizaciones que en esta se realicen. Esta librería facilita a su vez, la extracción de todo tipo de información acerca de los tuits publicados, así como de los usuarios que los publican, sus seguidores y usuarios seguidos entre otros.



Figura 3-30 Tweepy. [97]

Esta biblioteca proporciona al usuario una interfaz sencilla y fácilmente comprensible para mostrar y explicar las diferentes operaciones que con ella se pueden realizar. Uno de los grandes avances es la fácil forma de autenticación con la que obtener acceso a la API como se comprobará más adelante.

Para su instalación tan solo será necesario escribir el siguiente comando en el terminal de la máquina virtual (véase Figura 3-31):

```
maria@maria-VirtualBox:~$ pip3 install tweepy
```

Figura 3-31 Comando para instalar la librería de Python Tweepy. (Autoría propia)

3.5.3 Obtención de los 100 últimos tuits de un usuario y preprocesador de texto

A continuación, se procede a desarrollar el código que permita obtener los cien últimos tuits de los usuarios de los cuales se pretende determinar si pueden llegar a ser de interés para el ámbito de defensa. Para su obtención será necesario llevar a cabo el proceso esquematizado en la Figura 3-32.



Figura 3-32 Proceso para la obtención y limpieza de los últimos cien tuits de un usuario. (Autoría propia)

El código que procede a ejecutar lo indicado en el esquema anterior se muestra en la Figura 3-33. En este, se define la función *obtener_100ult* mediante la cual se obtienen los últimos cien tuits de una cuenta de Twitter cuyo nombre de usuario ha sido previamente escrito de manera íntegra y sin el '@'. Se excluyen los retuits y se imprime únicamente el texto del tuit, ignorando el resto de información acerca de este cuya relevancia no es clave para el desarrollo del programa.

```
1 import re  
2 import emoji  
3 from googletrans import Translator  
4 from langdetect import detect  
5 import tweepy  
6 import json  
7 translator = Translator()  
8  
9 bearer_token = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXJa"  
  
10  
11 def obtener_100ult(usuario):  
12     client = tweepy.Client(bearer_token=bearer_token, wait_on_rate_limit=True)  
13     query = f'from:{usuario} -is:retweet'  
14     tweets = client.search_recent_tweets(query=query, tweet_fields=['text'], max_results=100)  
15     print("Se han encontrado " + str(len(tweets.data)) + " tuits")  
  
16  
17     with open("/home/maria/Escritorio/100ult", "w") as cien:  
18         for tweet in tweets.data:  
19             cien.write(tweet.text)  
20  
21 usuario = input("Ingrese el nombre del usuario de Twitter (completo y sin '@'): ")  
22 obtener_100ult(usuario)  
23  
24 with open('/home/maria/Escritorio/100ult', 'r') as tuits_mal, open('/home/maria/Escritorio/tuits_buenos', 'a') as tuits_bien:  
25     tweets = tuits_mal.readlines()  
26     linea_final = ''  
27  
28     for linea in tweets:  
29         linea_limpia = re.sub(r'https?://\S+\s*', '', linea) # Eliminar los enlaces  
30         linea_limpia = re.sub(r'\@w+s*', '', linea_limpia) # Eliminar las menciones  
31         linea_limpia = re.sub(r'#(\w+)', '', linea_limpia) # Eliminar los hashtags  
32         linea_limpia = emoji.get_emoji_regexp().sub('', linea_limpia) # Elimina todos los emoticonos  
33         linea_limpia = linea_limpia.lower() # Convertir todo a minúsculas  
34         linea_final += linea_limpia.strip() + ' ' # Concatenar las líneas limpias separadas por un espacio  
35  
36 if len(linea_final.strip()) > 0: # Verificar si la línea contiene algún carácter antes de escribir  
37     if detect(linea_final) != 'es':  
38         linea_final = translator.translate(linea_final).text  
39     tweet_text = f'{"user_name": "{usuario}", "text": "{linea_final.strip()}"}, {"defensa": {0}}'  
40     print(tweet_text, file=tuits_bien, end='\n')  
41     tuits_bien.write("\n")
```

Figura 3-33 Código de *Python* para la obtención de los cien últimos tuits de un usuario y preprocesado de texto.
(Autoría propia)

Con el fin de posteriormente poder procesar de manera eficiente este fichero para la obtención de una representación vectorial del mismo (*embeddings*), se precisará en primer lugar la limpieza y el preprocesado de los tuits obtenidos. En este caso se procederá a un filtrado exhaustivo mediante las siguientes modificaciones:

- Eliminar enlaces.
- Eliminar menciones.
- Eliminar etiquetas.
- Eliminar emoticonos.
- Conversión de las mayúsculas a minúsculas.
- Traducir los tuits a inglés en caso de no encontrarse en este idioma.

Cabe destacar que este código aúna el texto de todos los tuits de un usuario en una misma línea e imprime en el fichero *tuits_buenos* las siguientes variables separadas por comas: ‘*user_name*’ (nombre de usuario), ‘*text*’ (contiene el texto de los cien últimos tuits), ‘*defensa*’ (etiqueta al usuario dentro del ámbito de defensa si vale 1, o no perteneciente a él cuando valga 0). Esta última variable se predefine manualmente para poder obtener los resultados de las predicciones y probabilidades de acierto y precisión a la hora de evaluar al clasificador (véase capítulo 4).

Previo a la ejecución del código será necesaria la instalación de:

- La versión 0.6.0 del paquete *emoji* de *Python* [98] empleando el gestor de paquetes para *Python* y mediante el comando de la Figura 3-34. Dicha biblioteca proporciona un modo sencillo de emplear y trabajar con los emoticonos.

```
maria@maria-VirtualBox:~$ pip3 install emoji==0.6.0
```

Figura 3-34 Comando para instalar la librería *emoji*. (Autoría propia)

- La librería de procesamiento de lenguaje natural *LangDetect* [99] (mediante el comando de la Figura 3-35) empleada para detectar de manera automática el idioma de un texto. Cabe destacar su capacidad de detección de más de 55 idiomas diferentes.

```
maria@maria-VirtualBox:~$ pip3 install langdetect
```

Figura 3-35 Comando para instalar la librería *LangDetect*. (Autoría propia)

- Finalmente, la versión 4.0.0-rc1 de la librería *Googletrans* [100] (mediante el comando de la Figura 3-36), la cual proporciona una interfaz sencilla para la API de Google Translate con el fin de traducir fácilmente texto de un idioma a otro.

```
maria@maria-VirtualBox:~$ pip3 install googletrans==4.0.0-rc1
```

Figura 3-36 Comando para instalar la librería *Googletrans*. (Autoría propia)

El código mostrado en la Figura 3-33 importa a su vez la biblioteca *re* [101] empleada para la detección de patrones en el texto y su eliminación mediante el empleo de expresiones regulares. La función '*sub ()*' es un método de la biblioteca '*re*' empleado para reemplazar una cadena de texto por otra. Esta función recibe tres parámetros; el primero corresponde a la cadena a sustituir, el segundo la cadena por la que se va a reemplazar y finalmente el último hace referencia a la cadena de entrada en la cual se va a buscar la indicada en el primer parámetro.

3.6 Implementación del clasificador y entrenamiento

La clasificación supervisada, como se ha expuesto previamente, es una rama perteneciente al *Machine Learning* en la que se encuadran algoritmos tales como: árboles de decisión, clasificadores de regresión logística, redes bayesianas, etc. El algoritmo seleccionado para crear el clasificador que permita llevar a cabo el sistema inteligente de detección, como se indicó previamente, es el algoritmo *Random Forest* (véase apartado 2.5.4).

3.6.1 Justificación de la elección del algoritmo

Este algoritmo ha sido seleccionado tras un exhaustivo estudio. En primer lugar, se ha analizado el problema que se pretende solventar: la creación de una herramienta de detección inteligente de usuarios de la red social Twitter que sean de interés por publicar información acerca del ámbito de la defensa. En definitiva, consiste en clasificar unos datos entrantes (tuits del usuario en cuestión) en dos salidas bien definidas (clasificador binario: Defensa=1 si pertenece a dicho ámbito o Defensa=0 si, por el contrario, no pertenece) y en base a unos datos que han sido previamente encasillados en una de las dos posibles etiquetas (corpus de artículos de Wikipedia). Es por ello que, en primer lugar, se ha de seleccionar un algoritmo de aprendizaje automático supervisado frente a uno no supervisado o semisupervisado. En cuanto al algoritmo a emplear, será óptimo encontrar uno apto para trabajar con un alto volumen de datos y obtener unas precisas predicciones. Es por lo que, entre las opciones barajadas descritas en el apartado 2.5.4, y debido a su simplicidad y alta capacidad de procesamiento [71], se ha seleccionado *Random Forest*.

Una de las desventajas que presentan los algoritmos supervisados es la gran cantidad de datos ya etiquetados que precisan en el entrenamiento para poder llevar a cabo una precisa clasificación. Este problema queda solventado con la base de datos creada: el corpus compuesto por aproximadamente 10.000 artículos de Wikipedia pertenecientes al ámbito de la defensa y 10.000 que no.

3.6.2 Corpus para el entrenamiento. Obtención de embeddings

La obtención de un corpus extenso es fundamental para llevar a cabo el proceso de entrenamiento del clasificador con el objeto de que pueda realizar una correcta clasificación de los futuros tuits de un usuario cualquiera. En este caso, esta base de datos es la extraída en el apartado 3.4 y volcada en el fichero *articulos_json*.

En primer lugar, hay que procesar este conjunto de datos como un diccionario de *Python*. Cabe recordar que los parámetros recogidos en este archivo correspondientes a cada artículo eran los siguientes:

- a) 'id': muestra el identificador interno del artículo.
- b) 'url': contiene la dirección web del artículo en cuestión.
- c) 'title': guarda el título del artículo.
- d) 'text': fragmento de texto extraído del artículo en cuestión.
- e) 'Defensa': etiqueta que encasilla el artículo como perteneciente o no a dicho ámbito.

En dicho diccionario de *Python* se va a excluir la variable 'url' puesto que no se considera necesarias para el correcto funcionamiento del clasificador. Se emplearán por tanto las siguientes en el orden a continuación mostrado: 'id', 'title', 'Defensa' y 'embedding'.

Las *embedding* (incrustaciones) de texto [102], son uno de los mayores avances alcanzados en la resolución de problemas de procesamiento del lenguaje natural (PLN). Consiste en una técnica de codificación basada en la representación de palabras, oraciones o documentos de texto, en la cual aquellas que compartan significado, o este sea similar, poseerán una representación similar. La principal finalidad es la representación del significado de cadenas de texto o caracteres mediante vectores de valor real (a menudo con cientos de dimensiones) en espacios vectoriales predefinidos. Esto permite a las máquinas procesar de manera efectiva el lenguaje humano. Además, el hecho de que palabras con significados similares obtengan una codificación similar permite una búsqueda más comprensiva y así lograr óptimos resultados. En este caso la creación de *embeddings* se realiza para emplearlas como entradas al clasificador para determinar similitudes entre fragmentos de texto, en base a las representaciones vectoriales, tanto ahora en la primera fase de entrenamiento, como más adelante para las diversas pruebas con tuits. Estas incrustaciones son también empleadas para búsquedas o consultas, detección de anomalías en un determinado documento o medición de la diversidad, entre otros.

Se empleará la API de OpenAI [78] para obtener dichos *embeddings*. Cabe destacar que para obtener el acceso a estas conversiones es necesario tener acceso a la API premium de pago, no siendo gratuito su uso. Todo ello será explicado en detalle en el apartado de repercusiones económicas.

Para la conversión del texto en *embeddings* se empleará el código mostrado en la Figura 3-38, tras haber instalado previamente las siguientes librerías con los comandos mostrados en la Tabla 3-5.

<i>Pandas</i>	pip3 install pandas	[103]
<i>OpenAI</i>	pip3 install openai	[104]
<i>Transformers</i>	pip3 install transformers	[105]
<i>Plotly</i>	pip3 install plotly	[106]
<i>Matplotlib</i>	pip3 install matplotlib	[107]
<i>Scikit-learn</i>	pip3 install scikit-learn	[108]
<i>Torch torchvision</i>	pip3 install torch torchvision	[109]
<i>SciPy</i>	pip3 install scipy	[110]

Tabla 3-5 Comandos para instalar las librerías desde el terminal. (Autoría propia)

El procedimiento para llevar a cabo este código se basa en este sencillo diagrama (véase Figura 3-37) en el cual se pueden ver representadas cuáles serían las entradas y cuáles las salidas de este.

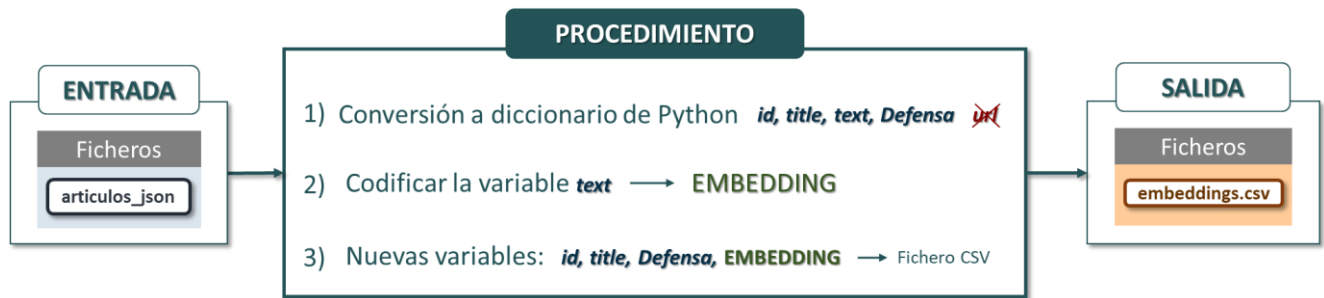


Figura 3-37 Proceso de obtención de *embeddings* de artículos de Wikipedia. (Autoría propia)

En base al ejemplo provisto en la página de *GitHub* [111], en el cual se emplea un *dataset*, compuesto por reseñas sobre comida recomendable de usuarios de Amazon clasificadas como positivas y negativas, que se codifica transformando el texto en un *embedding*; se desarrolla el código mostrado en la Figura 3-38 y esquematizado en la Figura 3-37.

```

import pandas as pd
import tiktoken
import openai
import csv
import time
from openai.embeddings_utils import get_embedding

embedding_model = "text-embedding-ada-002"
embedding_encoding = "cl100k_base"
max_tokens = 8000

openai.api_key = "sk-6hs"

with open('/home/maria/Escritorio/articulos', 'r') as art:
    lineas = art.readlines()

articulos = []

for linea in lineas:
    articulo = eval(linea)
    articulo['text'] = articulo['text'].replace('\n', '')
    del articulo['url']
    articulos.append(articulo)

encoding = tiktoken.get_encoding(embedding_encoding)

with open('/home/maria/Escritorio/embeddings.csv', 'a') as f:
    fieldnames = ['id', 'title', 'Defensa', 'embedding']
    writer = csv.DictWriter(f, fieldnames=fieldnames, delimiter=";", quotechar='"')
    writer.writeheader()

    for articulo in articulos:
        print("Procesando...", articulo['title'])
        embeddings = {}
        embeddings['id'] = articulo['id']
        embeddings['title'] = articulo['title']
        embeddings['Defensa'] = articulo['Defensa']
        encoded = encoding.encode(articulo['text'])

        if len(encoded) < max_tokens:
            embedding = get_embedding(articulo['text'], engine = embedding_model)
            if embedding is not None:
                embeddings['embedding'] = str(embedding)
                writer.writerow(embeddings)

        time.sleep(1)
        print('Obtenido embedding del artículo:', articulo['title'])
  
```

Figura 3-38 Código de *Python* para obtener *embeddings* de los artículos de Wikipedia y su almacenamiento en un archivo CSV. (Autoría propia)

Dicho código tiene como principal propósito la obtención de un archivo CSV con todos los artículos inicialmente contenidos en el fichero de texto *articulos_json* con sus respectivos *embeddings*. Para comenzar, se importan todas las librerías necesarias previamente instaladas. Cabe destacar el empleo de la librería Pandas para el manejo de los datos y la librería OpenAI para la obtención de los *embeddings* de texto. También se importa la función *'get_encoding ()'* de esta última librería. Tras esto, son establecidas las variables *embedding_model* (como modelo de *embedding* a emplear), *embedding_encoding* y *max_tokens* (indicando el máximo número de tokens que se permiten usar para la generación del *embedding*).

Mediante la clave de la API de OpenAI se puede acceder a los servicios necesarios de la plataforma. Se abre el archivo que contiene todos los artículos de Wikipedia necesarios para entrenar el clasificador y se itera sobre cada uno de ellos realizando lo siguiente: eliminar la variable *'url'* del diccionario de *Python* reemplazándola por cadenas vacías, se crea el diccionario *embeddings* para almacenar los resultados y mediante la función *'encode ()'* se obtiene la codificación de la variable *'text'*. Finalmente, si la longitud de la codificación es menor que el valor *max_tokens* se procede a lo siguiente (cabe destacar que se ha comprobado previamente que la longitud de cada *embedding* ronda las 1.500 posiciones): se genera el *embedding* utilizando la función *'get_embedding ()'* de OpenAI y si este se ha creado correctamente se almacena en el diccionario *'embeddings'* escribiendo una línea en el archivo CSV la cual contiene la siguiente información sobre cada artículo:

- a) *'id'*
- b) *'title'*
- c) *'Defensa'* (etiqueta con valor 0 o 1)
- d) *embedding* del texto

Se realiza una pausa de un segundo previo a continuar con el siguiente artículo y se muestra en pantalla un mensaje indicando que se ha obtenido el *embedding* del artículo en cuestión.

Cabe destacar que la ejecución de este código, pese a que la obtención de un *embedding* apenas dura dos segundos, puesto que es necesario la obtención del *embedding* de aproximadamente veinte mil artículos y el tiempo de espera entre cada obtención era de un segundo, ha requerido aproximadamente diez horas hasta la ejecución completa.

3.6.3 Entrenamiento del clasificador

Finalmente se emplea el fichero recientemente creado: *embeddings.csv* como base de datos para la obtención del clasificador y proceder a su entrenamiento. En la Figura 3-39 se muestra de manera esquemática el proceso que se va a seguir en el código desarrollado para dichos fines.

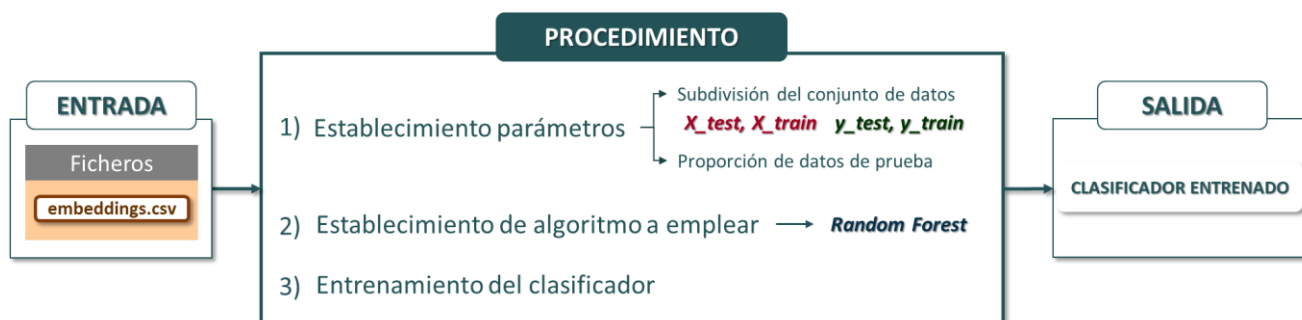


Figura 3-39 Proceso de obtención y entrenamiento del clasificador. (Autoría propia)

En base al ejemplo publicado en la página [112], se ha desarrollado el código mostrado en la Figura 3-40, el cual se enfoca en la creación de un modelo de clasificación de bosque aleatorio empleando el conjunto de datos del fichero *embeddings* creado, junto con sus respectivas etiquetas.

```

1  import pandas as pd
2  import numpy as np
3  from sklearn.ensemble import RandomForestClassifier
4  from sklearn.model_selection import train_test_split
5  from sklearn.metrics import classification_report, accuracy_score
6
7  fichero = "/home/maria/Escritorio/embeddings.csv"
8
9  fich = pd.read_csv(fichero, delimiter = ";")
10 print(fich.columns)
11 fich['embedding'] = fich.embedding.apply(eval).apply(np.array)
12
13 X_train, X_test, y_train, y_test = train_test_split(list(fich.embedding.values), fich.Defensa, test_size=0.2, random_state=42)
14
15 clf = RandomForestClassifier(n_estimators=100)
16
17 clf.fit(X_train, y_train)

```

Figura 3-40 Código de Python para entrenar el clasificador con el conjunto de datos de Wikipedia. (Autoría propia)

En primer lugar, importa las librerías necesarias: *pandas* para leer el archivo CSV y manipular datos tabulares, *numpy* para la manipulación de matrices y *scikit.learn* para entrenar y posteriormente evaluar el modelo de clasificación.

Tras esto, se procede a guardar en la variable *fichero* el archivo CSV obtenido previamente empleando para su lectura la función '*read_csv ()*' así como para crear un *DataFrame* de *pandas*. El argumento *delimiter* se emplea para indicar que las diferentes variables del fichero están separadas por ';'. Mediante la siguiente línea de código, se convierte y sustituye la columna *embedding* en una matriz *numpy*. Se procede entonces a la división del conjunto de datos en diferentes subconjuntos:

- *X_train*: Contiene las características de los datos de entrenamiento.
- *X_test*: Contiene las características de los datos de prueba.
- *y_train*: Contiene las etiquetas de los datos de entrenamiento
- *y_test*: Contiene las etiquetas de los datos de prueba.

Para ello se emplea la función '*train_test_split ()*' y se emplean '*list (fich.embedding.values)*' como la variable de características *X*, '*fich.Defensa*' se utiliza como la variable objetivo a predecir '*y*'. Mediante el parámetro *test_size* se indica la proporción de datos que se usarán para el conjunto de prueba (siendo en este caso un 20%) y finalmente *random_state* fija la semilla aleatoria para garantizar la reproducibilidad de los resultados obtenidos.

Se emplea la clase '*RandomForestClassifier*' con el fin de crear el modelo de clasificación en el cual se indica el empleo de cien árboles. La función '*fit ()*' es un método de '*RandomForestClassifier*' que ajusta y entrena el modelo utilizando el conjunto de características de entrenamiento: *X_train*, y sus respectivas etiquetas: *y_train*. El modelo aprende mediante esta línea a mapear las características del conjunto de datos de entrenamiento a una determinada etiqueta que indica a qué clase pertenece.

Una vez entrenado el clasificador, este está preparado para realizar predicciones sobre nuevos conjuntos de datos y evaluar su probabilidad de acierto.

En la siguiente Figura 3-41 se muestra, en líneas generales, el proceso que llevaría cabo la herramienta desarrollada en el presente proyecto. Este sistema inteligente recibirá el nombre del usuario de Twitter a estudiar, mediante los códigos previamente justificados se obtendrán los cien últimos tuits de dichos usuarios los cuales serán integrados en un mismo texto. Una vez obtenido un fichero en el cual se encuentran contenidos dichos tuits, se obtendrá un *embedding* codificando el texto entrante. A partir de estos *embeddings*, con el clasificador previamente entrenado, se procederá a catalogar al usuario en base a sus tuits en una clase u otra, es decir, perteneciente o no al ámbito de defensa.

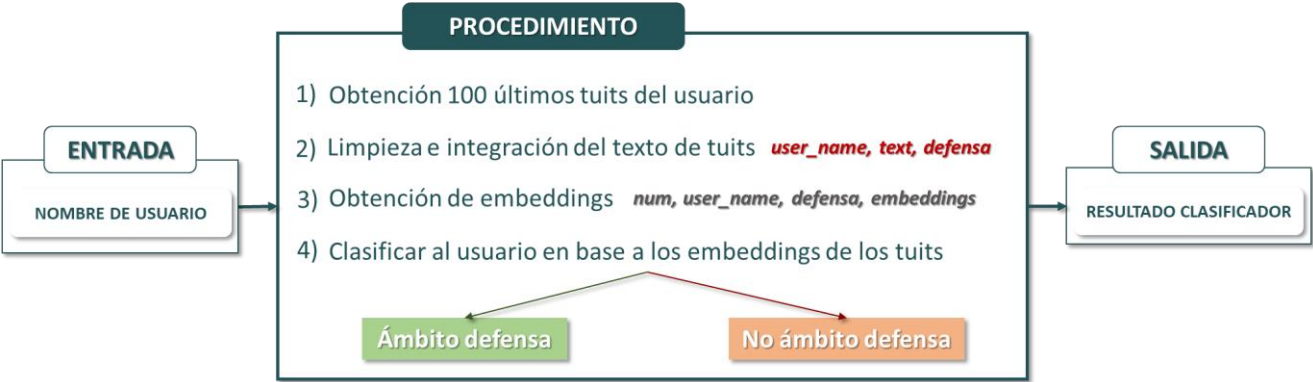


Figura 3-41 Proceso general del sistema de detección de usuarios pertenecientes al ámbito de defensa. (Autoría propia)

4 PRUEBA Y VALIDACIÓN

En el presente capítulo se procede, a presentar e interpretar los resultados obtenidos al emplear diferentes conjuntos de datos en el clasificador ya entrenado comenzando con el propio corpus de Wikipedia. Tras esta primera prueba, se obtendrán y evaluarán las predicciones que este realiza sobre el conjunto de datos a estudiar como objeto de este proyecto, compuesto por tuits de cuarenta usuarios: veinte de ellos englobados en el ámbito de defensa y veinte ajenos a este.

Una vez evaluados ambos corpus, se procederá a realizar un experimento mediante el cual se pretende introducir los tuits de cada uno de los usuarios en el *chatbot* de ChatGPT, a través del código que se mostrará más adelante, con el fin de ver cuán capaz es de clasificarlo como de interés para el ámbito de defensa o no.

Cabe destacar que previo a todo ello se hará una breve explicación del conjunto de métricas que se pretende emplear para el análisis de los resultados obtenidos.

4.1 Matriz de confusión y métricas empleadas para evaluar el clasificador

En la siguiente figura se muestra la matriz de confusión en la que se recogen los posibles resultados que pueden obtenerse en la salida del clasificador en base a la certeza y el ajuste que este hace con respecto al valor o etiqueta real de cada característica a evaluar (véase Figura 4-1) [113]:

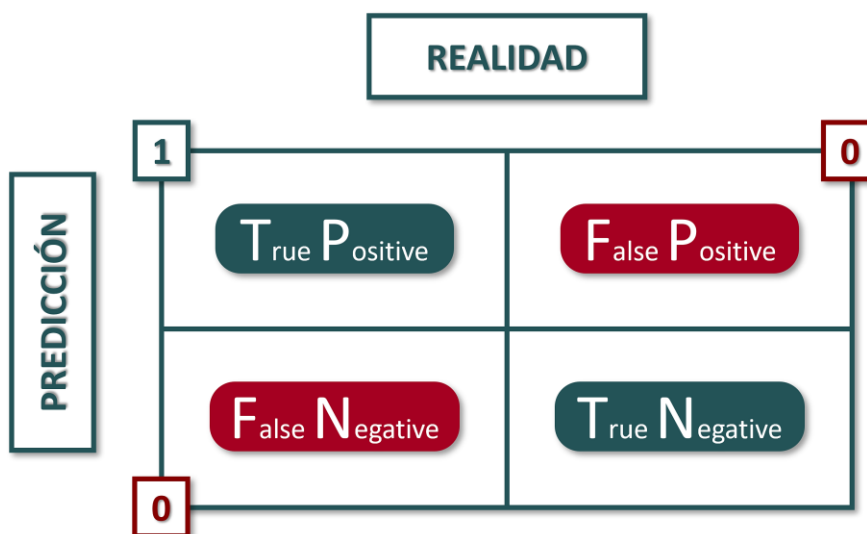


Figura 4-1 Matriz de confusión con posibles salidas del clasificador. (Autoría propia)

- *True Positives* (TP, Verdaderos positivos): La instancia pertenece al ámbito de defensa (1) y es clasificado correctamente (1).
- *True Negative* (TN, Verdaderos negativos): La instancia no pertenece al ámbito de defensa (0) y es clasificado como tal (0).
- *False Positive* (FP, Falso Positivo): La instancia no pertenece al ámbito de defensa (0) y es clasificado como positivo (1).
- *False Negative* (FN, Falso Negativo): La instancia pertenece al ámbito de defensa (1) y es clasificado como negativo (0).

Se establecen unas métricas cuya función es determinar cuán eficaz es el clasificador en base a dichas posibles clasificaciones. Estas son las siguientes [113]:

- La precisión: Mide la proporción de instancias clasificadas como positivas con las que realmente son positivas. Es decir, es una medida de exactitud que indica las correctamente clasificadas como pertenecientes al ámbito de defensa con respecto al total de positivas, incluyendo las clasificadas como tal siendo realmente no pertenecientes a dicho ámbito.

$$\text{Precisión} = \frac{TP}{TP + FP} \quad [4-1]$$

- La exhaustividad o recuperación (*recall*): Mide la proporción de instancias positivas que son correctamente identificadas por el modelo. Es la relación entre las positivas correctamente identificadas entre el total de positivas reales que existen en el conjunto de datos como se puede comprobar en la siguiente ecuación:

$$\text{Recuperación} = \frac{TP}{TP + FN} \quad [4-2]$$

La principal diferencia entre la precisión y la recuperación es que esta última aporta el rendimiento que tiene el clasificador en cuanto a la detección de falsos negativos se refiere; mientras que el primero indica el rendimiento respecto al total de positivos que se obtienen como resultado.

- La puntuación F1 (F1 score): Es la media armónica de la precisión y la exhaustividad mediante la cual se proporciona una medida equilibrada de la calidad de la clasificación.

$$\text{Puntuación F1} = 2 \times \frac{\text{Precisión} \times \text{Recuperación}}{\text{Precisión} + \text{Recuperación}} \quad [4-3]$$

- Exactitud (*accuracy*): Mediante este parámetro se extrae el porcentaje final de elementos clasificados correctamente con respecto al total de ellos, cabe destacar su generalizado empleo debido a su sencilla interpretación.

$$\text{Exactitud} = \frac{TP + TN}{TP + FP + FN + TN} \quad [4-4]$$

- La media aritmética (*macro-averaged*): Es la precisión promedio calculada para cada clase, sin tener en cuenta el desequilibrio de las mismas. Cada una de ellas tiene un peso igual, por lo que esta métrica trata a todas las clases por igual independientemente de su tamaño. Es decir, que esta métrica da el mismo peso a todas las clases.

- La media ponderada (*weighted-averaged*): En este parámetro se tiene en cuenta el desequilibrio de las clases mediante el cálculo de un promedio ponderado de la precisión para cada clase, donde los pesos son proporcionales al número de muestras en cada una de ellas.

Cabe destacar que estas métricas son aplicables a todas las clases (No defensa-0-/Defensa-1-).

4.2 Evaluación inicial del clasificador con artículos de Wikipedia

Se procede a la realización de una primera prueba para evaluar cómo de entrenado está el clasificador mediante la obtención de predicciones sobre el conjunto de datos de Wikipedia. Para ello se muestra en la Figura 4-2 el código desarrollado.

```

1  import pandas as pd
2  import numpy as np
3  from sklearn.ensemble import RandomForestClassifier
4  from sklearn.model_selection import train_test_split
5  from sklearn.metrics import classification_report, accuracy_score
6
7  fichero = "/home/maria/Escritorio/embeddings.csv"
8
9  fich = pd.read_csv(fichero, delimiter = ";")
10 print(fich.columns)
11 fich['embedding'] = fich.embedding.apply(eval).apply(np.array)
12
13 X_train, X_test, y_train, y_test = train_test_split(list(fich.embedding.values), fich.Defensa, test_size=0.2, random_state=42)
14
15 clf = RandomForestClassifier(n_estimators=100)
16
17 clf.fit(X_train, y_train)
18
19 preds = clf.predict(X_test)
20
21 pruebas = clf.predict_proba(X_test)
22 print(pruebas)
23
24 report = classification_report(y_test, preds)
25 print(report)

```

Figura 4-2 Código de *Python* para obtener los parámetros de las predicciones del clasificador ya entrenado con el conjunto de datos de Wikipedia. (Autoría propia)

Se emplea, en primer lugar, la función '*predict()*' donde se indica el conjunto de datos de prueba, *X_test*, mediante el modelo de clasificación previamente entrenado '*clf*'. Esta línea predice la etiqueta de la clase para cada una de las observaciones de dicho conjunto de datos y lo guarda en la variable *preds*.

En la siguiente línea en vez de hacer las predicciones sobre una etiqueta de clase, se predice la probabilidad de cada clase para cada una de las observaciones, es decir, calcula la probabilidad de que cada uno de los artículos pertenezca a defensa ('*Defensa*'=1) y la probabilidad de que no lo haga ('*Defensa*'=0). Dicha salida se guarda en la variable *pruebas* y se imprime por pantalla obteniendo el siguiente resultado (véase Tabla 4-1):

	Prob. Def=0	Prob. Def=1
Instancia 1	0,94	0,06
Instancia 2	0,85	0,15
Instancia 3	0,24	0,76
..		
Instancia n-2	0,12	0,88
Instancia n-1	0,02	0,98
Instancia n	0,95	0,05

Tabla 4-1 Probabilidad de pertenecer a cada clase para cada instancia. (Autoría propia)

Finalmente, se almacena en la variable *report* el resultado de emplear la función ‘*classification_report()*’ de *scikit-learn* para generar un informe del clasificador. El código imprime el informe de la Tabla 4-2 en pantalla como se puede observar a continuación:

	Precision	Recall	F1-score	Support
0	0,96	1	0,98	2021
1	0,99	0,95	0,97	1794

Accuracy			0,97	3815
Macro avg	0,98	0,97	0,97	3815
Weighted avg	0,97	0,97	0,97	3815

Tabla 4-2 Resultados de la evaluación del clasificador con el corpus de Wikipedia. (Autoría propia)

En primer lugar, imprime la matriz de probabilidad de predicciones la cual muestra las probabilidades que posee cada artículo de prueba del conjunto de datos de Wikipedia de pertenecer a una de las dos clases: 0 y 1. Cada una de las filas de la matriz representa una instancia o artículo y las columnas representan las clases 0 y 1 respectivamente. Véase un ejemplo: el primer artículo escogido de prueba predice que tiene un 94% de probabilidad de pertenecer a la clase 0 (es decir, de no pertenecer al ámbito de defensa) y un 6% de probabilidad de pertenecer a este mismo ámbito.

A continuación, procede con las métricas previamente explicadas. En este caso, la precisión para la clase 0 es del 96% mientras que para la clase 1 es del 99% lo que implica que el modelo desarrollado ha llevado a cabo una clasificación en la cual los falsos positivos son muy bajos, puesto que una alta tasa de precisión indica que el modelo tiene una baja tasa de falsos positivos y, por tanto, es bastante exacta.

A su vez un alto porcentaje obtenido en la recuperación indica que el modelo tiene una baja tasa de falsos negativos. En este caso para la clase 0 se obtiene un 100% lo que demuestra la inexistencia de falsos negativos para dicha clase en el escenario de prueba. En el caso de la pertenencia al ámbito de la defensa, se obtiene un 95% lo que implica la existencia de falsos negativos.

El F1-score hace una media armónica de ambos parámetros para obtener una medida general del funcionamiento del clasificador para cada una de las clases. En este preciso caso, aunque por tan solo un 1%, se comprueba que la clase 0 obtiene una mejor puntuación. Uno de los motivos más probables es la existencia de un mayor número de artículos que pertenecen a dicha clase en el corpus con el que se ha procedido tanto a entrenar (XX frente a yy) el clasificador como a la evaluación del mismo (2021 artículos frente a 1794).

A su vez, se puede comprobar cómo se obtiene un 97% prácticamente en el resto de las métricas calculadas: macro precisión promediada (el modelo ha obtenido una precisión promedio del 98%), *accuracy* (el modelo ha clasificado correctamente el 97% de los artículos) y media ponderada (teniendo en cuenta el número de artículos de cada clase ha obtenido una precisión ponderada del 97%).

4.3 Evaluación del clasificador con tuits de usuarios

Una vez realizada la primera prueba del clasificador entrenado mediante el conjunto de datos de Wikipedia, se procede a la evaluación y prueba de la herramienta desarrollada en el entorno objetivo del presente trabajo: la clasificación de perfiles de usuarios de Twitter a partir de los contenidos (tuits) que publican. Con este fin, se han seleccionado manualmente cuarenta usuarios de la red social Twitter: veinte de ellos pertenecientes al ámbito de defensa y veinte que no como se muestra en la siguiente Tabla 4-3. Cabe destacar que se han seleccionado ciertos usuarios, prestablecidos como no

pertenecientes al ámbito, asociados a diferentes diarios y noticieros que, a pesar de no ser su principal tópico, en ocasiones tratan temas relacionados con defensa (como NYTimes).

Usuarios Defensa=1		Usuarios Defensa=0	
Osinttechnical	sentdefender	BBCWorld	CFR_org
KyivIndependet	UAWeapons	elonmusk	NBA
WarMonitor3	CovertShores	YouTube	espn
TheStudyOfWar	thewarzonewire	nytimes	epw_in
PhillipsPOBrien	Aviation_intel	SCMPnews	ThePrintIndia
NATO	ELINTNews	WSJ	Greenpeace
MilitaryLandnet	usairforce	NatGeo	UNICEF
oryxspioenkop	DrRadchenko	ScienceNews	UNHumanRights
NOELreports	TDF_UA	NASA	EWTnews
DefenceU	WarInstitute	FIFacom	ZARA

Tabla 4-3 Usuarios de Twitter seleccionados para evaluar el clasificador. (Autoría propia)

Una vez seleccionados se procede a aplicar el código pertinente explicado previamente (véase Figura 3-33) para extraer los cien últimos tuits de cada uno de los usuarios mediante la función de la API de Twitter `'search_recent_tweets()'` y almacenarlos en el fichero `tuits_buenos`, cuyo formato se puede visualizar en la Figura 4-3.

```

16 'user_name': 'elintnews', 'text': "correction: "blackout" not "coverage""the media coverage of the serious security incident is unnecessary and
unhelpful now. the time has come to remove the blackout order, reveal the details to the public and decide on a decisive response against our
enemies who have crossed red lines." on israel for the news coming out later on today.: russian su-27 aircraft struck propeller of a us mq-9 reaper
drone, causing us forces to have to bring the mq-9 down in international waters, according to via : 'incident' involving reaper drone over black
sea: western military sources tell such things as you can see is like dropping a hand grenade. people who are of course far more knowledgeable and
informed than i seem to also hold a similar view.i think it's time to abandon bakhmut.. just saying, long overduea few announcements to look for at
the next ukraine contact group meeting on wednesday: after the last meeting there were reports this next one may focus on aircraft, hopefully
croatia will announce 14 mi-8 helicopters for ukraine and poland and slovakia will announce mig-29 jetsi love how practically everyone replying to
this haven't even been to ukraine, yet they somehow know better than the guy who has been on the ground in bakhmut in the past few weeks and
regardless has studied the russian military for years and this war, in-depth, since day one. he's austrianbrother i'm not the one supporting the
invasion of ukraine lol my deepest condolences alina, very sadgeorgia should liberate abkhazia and south ossetia as soon as possible don't think
it's grass? the paint may have been applied we shall see anywaythink it might be carrying 6, if you look down the line at 0:09especially if a few
hundred more bradley ifv's are donated which i hope soeveryone's concentrating on ukraine's tanks but i'm more excited for 109 m2a2 bradley ifv's,
50 cv90 ifv's and 40 marder ifv's which i think may end up being more decisive. we will see.but importantly 34 out of 48 kh-101, kh-555 and kalibr
cruise missiles. the 8 kh-59 and kh-31 missiles missed their targets. the remaining 6 kh-22 hypersonic cruise missiles, 6 kinzhal air-launched
hypersonic ballistic missiles and 13 s-300 missiles can't be intercepted", 'defensa': 1
17 'user_name': 'usairforce', 'text': "- hosted a women's leadership symposium! more than 350 airmen took part in the hybrid event, which featured
senior leaders from the command, as well as from air staff and industry. is reinstating the enlisted college loan repayment program. potential
recruits, regardless of air force specialty code, may be eligible to apply for eclrp, which offers repayment plans up to $65,000. 's a highlight
from 's remarks during last week's women's air & space power symposium. the airshow in australia, the featured numerous usaf assets including
the b-52h stratofortress, c-17 globemaster iii, f-22 raptor, c-5m super galaxy & many more! department of the air force unveiled a $259.3
billion budget request designed to continue modernizing the air force & space force to meet evolving threats while also nourishing current needs
that include training, readiness & fostering new technology. foray into digital modernization took a leap forward when experts came together to
tackle new developmental test programs. helps support active, guard & reserve front-line airmen by showcasing the unique strengths,
opportunities - and importantly, the flexibilities - of each component of our within one fully-integrated air force. service members who are soon
to retire or separate from the military, skillbridge is an integral program that aids in transition & helps bridge the gap between end-of-
service and the beginning of civilian careers. fighter wing airmen, the federal aviation administration, & others successfully accomplished the
next step in an initiative that is building the premier east coast training environment to prepare combat aircrew for current & future combat
ops. - the point barrow long range radar site is a key node of our long-range radar system. .& air force personnel from brooke army medical
center's department of emergency medicine recently established a simulation training platform to increase readiness & meet joint comission
requirements for staff development and training. signed a memorandum, jan. 20, directing the stand-up of the f-22 raptor formal training unit at
joint base langley-eustis, virginia. - spoke at where he underscored the importance of our most valuable resource - our people. a high note from
's keynote address at this week. with allies & partners makes us stronger & helps ensure air superiority around the globe. recently had
the honor of training lebanon's 1st female fighter pilot! - throughout the upcoming months, a barriers to service cross-functional team is
examining existing policies & procedures to ensure they reflect the service members needed for the future.". 'defensa': 1

```

Figura 4-3 Fichero `tuits_buenos`. (Autoría propia)

Este fichero será la entrada del código que se mostrará a continuación y que se ha esquematizado en la Figura 4-4.

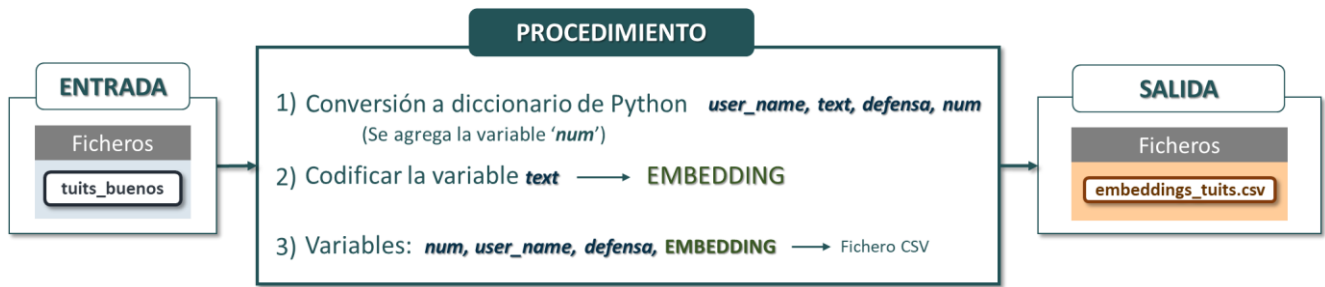


Figura 4-4 Proceso de obtención de *embeddings* de los tuits publicados por usuarios seleccionados. (Autoría propia)

Cabe destacar, en primer lugar, que este código (véase Figura 4-5) es similar al desarrollado para la obtención de los *embeddings* de los artículos de Wikipedia empleados para tanto el entrenamiento como la prueba inicial, con ciertas modificaciones que a continuación se explicarán.

```

1  import pandas as pd
2  import tiktoken
3  import openai
4  import csv
5  import time
6  from openai.embeddings_utils import get_embedding
7
8  embedding_model = "text-embedding-ada-002"
9  embedding_encoding = "cl100k_base"
10 max_tokens = 8000
11 openai.api_key = "s [REDACTED]"
12
13 with open('/home/maria/Escritorio/tuits_buenos', 'r') as tuits:
14     lineas = tuits.readlines()
15     tuits = []
16     num = 1
17     for linea in lineas:
18         if "'text':" in linea:
19             indice_texto = linea.find("'text': \"")
20             indice_defensa = linea.find("\", 'defensa':")
21             texto = linea[indice_texto+len("'text': \"):indice_defensa]
22             texto_modificado = texto.replace("'", '').replace('"', '').replace(',', '')
23             linea_modificada = linea[:indice_texto+len("'text': \")] + texto_modificado + linea[indice_defensa:]
24         else:
25             linea_modificada = linea
26         linea_modificada_con_llaves = '{' + linea_modificada.strip() + '}'
27         tuit = eval(linea_modificada_con_llaves)
28         tuit['num'] = num # add the 'numero' variable to the dictionary
29         num += 1 # increment the 'numero' variable
30         tuits.append(tuit)
31     encoding = tiktoken.get_encoding(embedding_encoding)
32
33     with open('/home/maria/Escritorio/embeddings_tuits.csv', 'a') as f:
34         fieldnames = ['num', 'user_name', 'defensa', 'embedding']
35         writer = csv.DictWriter(f, fieldnames=fieldnames, delimiter=";", quotechar='')
36         writer.writeheader()
37
38         for tuit in tuits:
39             print("Procesando...", tuit['num'])
40             embeddings = {}
41             embeddings['num'] = tuit['num']
42             embeddings['user_name'] = tuit['user_name']
43             embeddings['defensa'] = tuit['defensa']
44             encoded = encoding.encode(tuit['text'])
45             if len(encoded) < max_tokens:
46                 embedding = get_embedding(tuit['text'], engine = embedding_model)
47                 if embedding is not None:
48                     embeddings['embedding'] = str(embedding)
49                     # print(len(embedding)) # 1536
50                     writer.writerow(embeddings)
51
52             time.sleep(1)
53             print('Obtenido embedding de los tuits del usuario:', tuit['user_name'])
  
```

Figura 4-5 Código de *Python* para la obtención de los *embeddings* de los tuits. (Autoría propia)

Uno de los puntos más importantes a considerar a la hora de proceder al entrenamiento y posterior evaluación de un clasificador, es la estructura de los datos empleados para ambos casos. Es por ello, que se ha tenido que reorganizar la estructura del diccionario creado con los tuits de cada uno de los

usuarios seleccionados de tal manera que coincidan las variables de la siguiente manera (véase Tabla 4-4):

Corpus Wikipedia	<i>id</i>	<i>title</i>	<i>Defensa</i>	<i>embedding</i>
Corpus tuits	<i>num</i>	<i>user_name</i>	<i>defensa</i>	<i>embedding</i>

Tabla 4-4 Estructura de las variables contenidas en los diccionarios empleados en el clasificador. (Autoría propia)

Cabe destacar que se ha incorporado la nueva variable ‘num’ con el objetivo de numerar la lista de usuarios. Esta comienza valiéndose uno y va incrementando con el número de usuarios y sus respectivos tuits que va leyendo del fichero *tuits_buenos*.

Se procede, igual que con los artículos de Wikipedia, a codificar el texto y obtener los *embeddings*, en este caso de los tuits, con el objetivo de emplearlos como entrada en la evaluación del clasificador. Se realizará mediante la ejecución del código el cual irá indicando cómo progresa la ejecución y obtención mediante mensajes que se imprimen en pantalla, como se muestra en la Figura 4-6.

```

Procesando... 1
Obtenido embedding de los tuits del usuario: osinttechnical
Procesando... 2
Obtenido embedding de los tuits del usuario: kylindependent
Procesando... 3
Obtenido embedding de los tuits del usuario: warmonitor3
Procesando... 4
Obtenido embedding de los tuits del usuario: thestudyofwar
Procesando... 5
Obtenido embedding de los tuits del usuario: PhillipsPOBrien
Procesando... 6
Obtenido embedding de los tuits del usuario: nato
Procesando... 7
Obtenido embedding de los tuits del usuario: militarylandnet
Procesando... 8
Obtenido embedding de los tuits del usuario: oryxspioenkop
Procesando... 9
Obtenido embedding de los tuits del usuario: noelreports
Procesando... 10
Obtenido embedding de los tuits del usuario: defenceu
Procesando... 11
Obtenido embedding de los tuits del usuario: sentdefender
Procesando... 12
Obtenido embedding de los tuits del usuario: uaweapons
Procesando... 13
Obtenido embedding de los tuits del usuario: covertshores
Procesando... 14
Obtenido embedding de los tuits del usuario: thewarzonewire
Procesando... 15
Obtenido embedding de los tuits del usuario: aviation_intel
Procesando... 16
Obtenido embedding de los tuits del usuario: elintnews
Procesando... 17
Obtenido embedding de los tuits del usuario: usairforce
Procesando... 18
Obtenido embedding de los tuits del usuario: drradchenko
Procesando... 19
Obtenido embedding de los tuits del usuario: tdf_ua
Procesando... 20
Obtenido embedding de los tuits del usuario: warinstitute
Procesando... 21
Obtenido embedding de los tuits del usuario: bbcworld
Procesando... 22
Obtenido embedding de los tuits del usuario: elonmusk
Procesando... 23
Obtenido embedding de los tuits del usuario: youtube
Procesando... 24
Obtenido embedding de los tuits del usuario: nytimes
Procesando... 25
Obtenido embedding de los tuits del usuario: scmpnews
Procesando... 26
Obtenido embedding de los tuits del usuario: wsj
Procesando... 27
Obtenido embedding de los tuits del usuario: natgeo
Procesando... 28
Obtenido embedding de los tuits del usuario: sciencenews
Procesando... 29
Obtenido embedding de los tuits del usuario: nasa
Procesando... 30
Obtenido embedding de los tuits del usuario: fifacom
Procesando... 31
Obtenido embedding de los tuits del usuario: cfr_org
Procesando... 32
Obtenido embedding de los tuits del usuario: nba
Procesando... 33
Obtenido embedding de los tuits del usuario: espn
Procesando... 34
Obtenido embedding de los tuits del usuario: epw_in
Procesando... 35
Obtenido embedding de los tuits del usuario: theprintindia
Procesando... 36
Obtenido embedding de los tuits del usuario: greenpeace
Procesando... 37
Obtenido embedding de los tuits del usuario: unicef
Procesando... 38
Obtenido embedding de los tuits del usuario: unhumanrights
Procesando... 39
Obtenido embedding de los tuits del usuario: ewtnews
Procesando... 40
Obtenido embedding de los tuits del usuario: zara

```

Figura 4-6 Obtención de *embeddings*. (Autoría propia)

Una vez obtenidos los *embedding* correspondientes a los cien últimos tuits publicados por los usuarios elegidos, se inicia la evaluación definitiva de la herramienta metiéndole como entrada el archivo CSV *embeddings_tuits*. Se muestra el código para llevar a cabo dicha tarea en la siguiente imagen (véase Figura 4-7)

```

1 import pandas as pd
2 import numpy as np
3 from sklearn.ensemble import RandomForestClassifier
4 from sklearn.model_selection import train_test_split
5 from sklearn.metrics import classification_report, accuracy_score
6
7 fichero = "/home/maria/Escritorio/embeddings.csv"
8
9 fich = pd.read_csv(fichero, delimiter = ";")
10 print(fich.columns)
11 fich['embedding'] = fich.embedding.apply(eval).apply(np.array)
12
13 X_train, X_test, y_train, y_test = train_test_split(list(fich.embedding.values), fich.Defensa, test_size=0.2, random_state=42)
14
15 clf = RandomForestClassifier(n_estimators=100)
16
17 clf.fit(X_train, y_train)
18
19 fichero_tuits = "/home/maria/Escritorio/embeddings_tuits.csv"
20
21 fich_tuits = pd.read_csv(fichero_tuits, delimiter = ";")
22
23 fich_tuits['embedding'] = fich_tuits.embedding.apply(eval).apply(np.array)
24
25 tuits=[]
26 etiquetas_tuits=[]
27 tuits = list(fich_tuits['embedding'])
28 etiquetas_tuits=list(fich_tuits['defensa'])
29
30 preds_tuits = clf.predict(tuits)
31
32 pruebas_tuits = clf.predict_proba(tuits)
33 print(pruebas_tuits)
34
35 report_tuits = classification_report(etiquetas_tuits, preds_tuits)
36 print(report_tuits)

```

**Figura 4-7 Código de Python para evaluar el clasificador entrenado mediante el fichero *embeddings_tuits.csv*.
(Autoría propia)**

Se realiza el mismo proceso que el llevado a cabo con los artículos de Wikipedia entrenando primero el clasificador con el corpus de Wikipedia (*clf*) y que luego se emplea para clasificar los tuits, con la diferencia de las variables de características y etiquetas sobre las que se hace el estudio. En este caso se pretende predecir sobre el texto de los diferentes tuits que se almacenan en la lista *tuits* mientras que el informe se obtendrá en base a las predicciones y a la etiqueta añadida manualmente que indica la pertenencia real de dicho usuario al ámbito de la defensa o no.

Se muestran los resultados obtenidos de la ejecución de este código en la siguiente Tabla 4-5:

	Prob. Def=0	Prob. Def=1
Instancia 1	0,09	0,92
Instancia 2	0,11	0,89
Instancia 3	0,23	0,77
Instancia 4	0,16	0,84
Instancia 5	0,20	0,80
Instancia 6	0,15	0,86
Instancia 7	0,03	0,97
Instancia 8	0,21	0,79
Instancia 9	0,08	0,92
Instancia 10	0,12	0,88
Instancia 11	0,10	0,90
Instancia 12	0,05	0,95
Instancia 13	0,20	0,80
Instancia 14	0,04	0,96
Instancia 15	0,21	0,79
Instancia 16	0,23	0,77

	Prob. Def=0	Prob. Def=1
Instancia 21	0,41	0,59
Instancia 22	0,73	0,27
Instancia 23	0,85	0,15
Instancia 24	0,80	0,20
Instancia 25	0,49	0,51
Instancia 26	0,71	0,29
Instancia 27	0,88	0,12
Instancia 28	0,95	0,05
Instancia 29	0,87	0,13
Instancia 30	0,80	0,20
Instancia 31	0,71	0,29
Instancia 32	0,66	0,34
Instancia 33	0,58	0,43
Instancia 34	0,87	0,13
Instancia 35	0,67	0,33
Instancia 36	0,89	0,11

Instancia 17	0,42	0,58		Instancia 37	0,89	0,11
Instancia 18	0,41	0,59		Instancia 38	0,80	0,20
Instancia 19	0,10	0,90		Instancia 39	0,90	0,10
Instancia 20	0,39	0,61		Instancia 40	0,71	0,29

	Precision	Recall	f1-score	Support
0	1,00	0,90	0,95	20
1	0,91	1,00	0,95	20

Accuracy			0,95	40
Macro avg	0,95	0,95	0,95	40
Weighted avg	0,95	0,95	0,95	40

Tabla 4-5 Resultados obtenidos de la evaluación del clasificador mediante tuits de 40 usuarios. (Autoría propia)

La validación que se puede extraer a primera vista mediante la primera matriz impresa como resultado es que, teniendo en cuenta que las primeras veinte líneas corresponden a los primeros veinte usuarios seleccionados como pertenecientes al ámbito en cuestión, las probabilidades de pertenencia o no, es decir de clasificación del usuario en la clase 0/1, se ajustan bastante a la realidad. Pues, en la primera fila de la matriz, por ejemplo, se puede observar cómo existe un 91,5% de probabilidad de que el usuario Osinttechnical pertenezca al ámbito de defensa frente a un 8,5% de que no pertenezca.

Esta evaluación, pese a obtener métricas más bajas que en la anterior, obtiene unos resultados bastante certeros. Comenzando por el análisis de la clase 0, la cual ha obtenido una precisión del 100% y un *recall* del 90% lo que indica una tasa muy baja de falsos negativos y la nula existencia de falsos positivos lo que implica que ninguna instancia/usuario clasificada como perteneciente al ámbito de defensa fue erróneamente clasificada como no perteneciente.

En cuanto al conjunto de usuarios pertenecientes a la clase 1, se pueden observar coherentemente resultados prácticamente inversos pues se obtiene una precisión del 91% lo que implica que un 9% de los usuarios que no pertenecían al ámbito fueron clasificados como pertenecientes a la clase 1. Por otro lado, se obtiene un 100% de recuperación lo que indica que no se produjo ningún falso negativo, es decir, todos los usuarios pertenecientes al ámbito de defensa fueron clasificados como tal.

Se puede concluir un buen desempeño del modelo en general gracias a la puntuación F1 que así lo demuestra con un valor del 95%. Esta coincide con la *accuracy*, la cual afirma que el 95% de los usuarios fueron correctamente clasificados lo que implica que tan solo 2 usuarios se catalogaron en una clase que no les correspondía.

En definitiva, se puede comprobar cómo el clasificador se ajusta de una manera general y razonable en sus predicciones a la realidad. En este conjunto de datos de cuarenta usuarios se ha obtenido en líneas generales un 95% de acierto lo que supone tan solo dos usuarios mal clasificados y, por tanto, una clasificación bastante aceptable.

Cabe destacar que los dos usuarios erróneamente clasificados son @SCMPNews, usuario que publica noticias relacionadas con Hong Kong, China y Aisa en general; y @BBCWorld, cuenta cuya principal función es la publicación de noticias, reportajes y análisis del mundo. Su confusión puede quedar justificada pues, al tratarse de un servicio de difusión de noticias, publica tuits con la temática de defensa y las diferentes guerras como tema de actualidad, entre otros, diariamente.

4.4 Evaluación de respuestas obtenidas en ChatGPT

ChatGPT, como bien se ha expuesto previamente, consiste en un *chatbot* diseñado para dar respuesta a preguntas introducidas por pantalla por cualquier cliente que haga uso de él. Es por esta

razón por la que se ha optado por su empleo para que lleve a cabo la misma tarea de clasificación que la realizada por el clasificador entrenado en los anteriores apartados. Este *chatbot*, contiene una inmensa base de datos y está diseñado para que vaya aprendiendo de manera automática lo que en un principio debería posibilitar su capacidad para clasificar a un usuario, en base a sus últimos tuits, en el ámbito de la defensa o no.

Surge la idea de *Prompt Engineering* [114], pues para obtener respuestas lo más acordes y coherentes posibles, será preciso realizar una pregunta adecuada. *Prompt Engineering* tiene como objetivo principal la optimización de la calidad y el rendimiento ofrecido mediante un exhaustivo ajuste de la entrada del algoritmo o modelo de inteligencia artificial, aplicado al *chatbot* sería la pregunta a realizar. Para ello, no solo basta con emplear un lenguaje adecuado sino ser consciente del modo en el que interactúa y aprende el chat en cuestión, así como crear un contexto o antesala.

Teniendo todo ello en cuenta, se ha desarrollado el siguiente código cuyo proceso queda representado mediante el siguiente esquema (véase Figura 4-8):

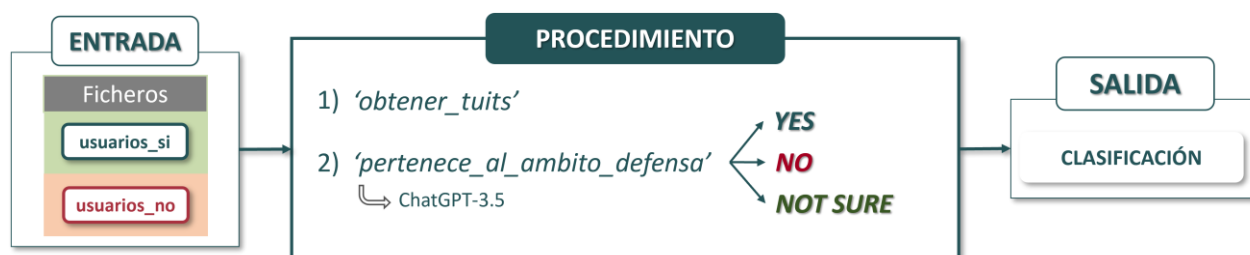


Figura 4-8 Proceso de clasificación mediante ChatGPT-3.5. (Autoría propia)

En este, en primer lugar, se define la función ya empleada anteriormente *'obtener_tuits ()'* la cual recibe como entrada el nombre de usuario. Sin embargo, en este caso en vez de obtener los cien últimos, tan solo se considera una muestra con los quince últimos tuits de cada usuario. Se define a su vez una segunda función denominada *'pertenece_al_ambito_defensa ()'* la cual recibe como único parámetro una lista de tuits, hace un preprocesamiento para evitar que aparezca información irrelevante para la clasificación (por ejemplo, URL) y emplea el modelo de lenguaje GPT-3.5 de OpenAI para llevar a cabo la clasificación. Comienza introduciendo al *chatbot* cuál es su misión y preguntándole cuál es la clasificación del usuario en base a los tuits extraídos, mediante las siguientes sentencias (véase Figura 4-9):

- “You are a chatbot that should answer my questions with either ‘yes’, ‘no’ or ‘not sure’.”
- “I will send you several tweets posted by a Twitter user. You have to classify the user based on whether their tweets belong to the defense field or not.”
- “Is the subject of the following text related to one or several of the following topics: defense, military, war, security or weapons?”

```

import openai
import re
import emoji
import tweepy
import json
bearer_token = "ja"

def obtener_tuits(usuario):
    client = tweepy.Client(bearer_token=bearer_token, wait_on_rate_limit=True)
    query = f"from:{usuario} -is:retweet"
    tweets = client.search_recent_tweets(query=query, tweet_fields=['text'], max_results=15)

    tuits = []
    if tweets is not None and tweets.data is not None:
        print("Se han encontrado " + str(len(tweets.data)) + " tuits")
        for tweet in tweets.data:
            tuits.append(tweet.text)
    return tuits

def pertenece_al_ambito_defensa(tuits):
    tuits_bien = []

    for linea in tuits:
        linea_limpia = re.sub(r"https?://\S+", "", linea)
        linea_limpia = re.sub(r"@w+\s+", "", linea_limpia)
        linea_final = emoji.get_emoji_regexp().sub("", linea_limpia)
        if linea_final.strip():
            tuits_bien.append(linea_final)

    tuits = " ".join(tuits_bien)
    openai.api_key = "ja"
    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        messages=[
            {"role": "system", "content": "You are a chatbot that should answer my questions with either 'yes', 'no' or 'not sure'"},
            {"role": "user", "content": "I will send you several tweets posted by a Twitter user. You have to classify the user based on whether their tweets belong to the defense field or not."},
            {"role": "user", "content": "To belong to the defense field he will have to talk only about defense, military, war, security or weapons related topics. If it talks about anything else, it is not related."},
            {"role": "user", "content": "Is the subject of the following text related to one or several of the following topics: defence, military, war, security, weapons? "},
            {"role": "user", "content": tuits }
        ]
    )
    result = ''
    for choice in response.choices:
        result += choice.message.content
    return result

with open('/home/maria/Escritorio/usuarios_si', 'r') as us_si:
    usuarios_si = us_si.readlines()

for usuario in usuarios_si:
    print("Analizando usuario:", usuario)
    tweets = obtener_tuits(usuario)
    if len(tweets) > 0:
        pertenece = pertenece_al_ambito_defensa(tweets)
        print("¿Podría resultar de interés para el ámbito de la defensa el usuario de Twitter", usuario, "en base a sus tuits?", pertenece)
    else:
        print("No se han podido encontrar tuits del usuario", usuario)

```

Figura 4-9 Código de Python para clasificar a los usuarios mediante ChatGPT. (Autoría propia)

Una vez definidas ambas funciones, el programa procede a abrir los respectivos ficheros en los que se encuentran contenidos los nombres de usuarios seleccionados en el apartado anterior (*usuarios_si* y *usuarios_no*) extrayendo los quince últimos tuits de cada uno de ellos y solicitando a ChatGPT-3.5 una clasificación en base a dichas publicaciones. Se obtienen las respuestas mostradas en la siguiente Tabla 4-6:

Usuarios Defensa=1		Usuarios Defensa=0	
Osinttechnical	Related	BBCWorld	Not sure
KyivIndependet	Related	elonmusk	No related
WarMonitor3	Related	YouTube	No related
TheStudyOfWar	Related	nytimes	No related
PhillipsPOBrien	Not sure	SCMPnews	No related
NATO	Related	WSJ	Not sure
MilitaryLandnet	Related	NatGeo	No related
oryxspioenkop	Not sure	ScienceNews	No related
NOELreports	Related	NASA	No related
DefenceU	Related	FIFacom	No related
sentdefender	Related	CFR_org	Not sure
UAWeapons	Related	NBA	No related
CovertShores	Not sure	espn	No related
thewarzonewire	Related	epw_in	No related

Aviation_Intel	Related	ThePrintIndia	No related
ELINTNews	Related	Greenpeace	No related
usairforce	Related	UNICEF	Not sure
DrRadchenko	No related	UNHumanRights	No related
TDF_UA	Related	EWTnews	No related
WarInstitute	Related	ZARA	No related

Tabla 4-6 Clasificación realizada por ChatGPT-3.5. (Autoría propia)

En base a dicha clasificación se obtienen las siguientes métricas para la clase 1 (cabe destacar, que aquellos clasificados como ‘*not sure*’ serán tomados como falsos negativos y falsos positivos respectivamente pues no se sabido determinar su definitiva clasificación):

$$Precisión_1 = \frac{TP}{TP + FP} = \frac{16}{16 + 4} = 0.8 \quad [4-5]$$

$$Recuperación_1 = \frac{TP}{TP + FN} = \frac{16}{16 + 4} = 0,8 \quad [4-6]$$

$$Puntuación F1_1 = 2 \times \frac{Precisión_1 \times Recuperación_1}{Precisión_1 + Recuperación_1} = 2 \times \frac{0,8 \times 0,8}{0,8 + 0,8} = 0,8 \quad [4-7]$$

En cuanto a la clase 0 se refiere, los resultados obtenidos serían los siguientes:

$$Precisión_0 = \frac{TP}{TP + FP} = \frac{16}{16 + 4} = 0,8 \quad [4-8]$$

$$Recuperación_0 = \frac{TP}{TP + FN} = \frac{16}{16 + 4} = 0,8 \quad [4-9]$$

$$Puntuación F1_0 = 2 \times \frac{Precisión_0 \times Recuperación_0}{Precisión_0 + Recuperación_0} = 2 \times \frac{0,8 \times 0,8}{0,8 + 0,8} = 0,8 \quad [4-10]$$

La exactitud también se mantiene constante en ambas clases:

$$Exactitud = \frac{TP + TN}{TP + FP + FN + TN} = \frac{16 + 16}{16 + 4 + 4 + 16} = 0.8 \quad [4-11]$$

En vista de los resultados obtenidos, se puede comprobar que se obtienen resultados ligeramente mejores mediante el método desarrollado en el presente trabajo, pues los obtenidos mediante el *chatbot* de OpenAI tienen un mayor porcentaje final de usuarios clasificados erróneamente lo que implica más fallos en sus predicciones. De cualquier manera, es preciso aclarar que se ha tratado de un primer experimento para analizar la viabilidad de este enfoque alternativo de clasificación, y dado su carácter limitado, y teniendo en cuenta que se podrían explorar ajustes en el *prompt* empleado o incluso ajustar (*fine tuning*) el modelo para la tarea de clasificación a realizar sería necesario extender estos experimentos antes de sacar conclusiones definitivas.

5 CONCLUSIONES Y LÍNEAS FUTURAS

5.1 Conclusiones

El impacto de la innovadora tecnología derivada de la Cuarta Revolución Industrial, en la cual el mundo se encuentra inmerso, ha alcanzado proporciones significativas. Tal es así, que se han producido numerosas obras cinematográficas en las que se representan exagerados avances de inteligencia artificial, en los cuales se pretende asemejar de manera más que razonable las máquinas a los seres humanos. Aunque aún, a día de hoy, se diste bastante de alcanzar tales aplicaciones de esta nueva tecnología e inteligencia artificial, son innegables los importantes progresos que en estas se están produciendo como bien ha quedado demostrado a lo largo del presente proyecto. Estas tecnologías tienen como misión principal facilitar la vida del ser humano en muchos aspectos, pues tareas que para estos puedan ser inabarcables por la cantidad de información que procesar (*Big Data*) entre otros, las máquinas son capaces de realizarlas con tan solo la introducción de unos algoritmos e instrucciones.

Esto último ha sido el punto de partida del presente proyecto, el *Machine Learning*. En base a este tipo de metodología y algoritmos, este trabajo tenía como objetivo principal la creación de un sistema que posibilitase la detección inteligente de usuarios de la red social Twitter que aportasen información que pudiese resultar de utilidad para el ámbito de la defensa. Con esta finalidad, se ha implementado una metodología, desarrollada minuciosamente en el tercer capítulo, mediante la cual poder llevar a cabo la creación de dicha herramienta.

Teniendo en cuenta cuáles eran los objetivos de partida y en base a los resultados ilustrados en el capítulo 1, se puede concluir la consecución en su totalidad del principal objetivo mediante la creación, entrenamiento y posterior evaluación de un clasificador, que permite discernir entre usuarios que publican en el ámbito de interés y los que no. Todo ello, mediante la extracción y el análisis de los últimos tuits que estos publicaron en la plataforma.

Se puede concluir, por tanto, la obtención de un relativamente preciso sistema de catalogación del usuario de la red social dentro del ámbito de defensa o ajeno a él.

A su vez, cabe destacar la implementación que se ha llevado a cabo con el fin de realizar una comparativa entre el sistema de detección inteligente, desarrollado en el vigente proyecto, y la clasificación que es capaz de obtener el recientemente afamado *chatbot* lanzado por la organización OpenAI en el pasado mes de noviembre y basado en su tecnología GPT-3.5: ChatGPT. Como conclusión de esta comparativa, y en base a los resultados obtenidos en ambas metodologías se puede afirmar que el sistema de detección inteligente desarrollado logra una mayor precisión en el escenario considerado. Esto se puede deber a que, en cierto modo, para lograr buenas y más certeras respuestas a las preguntas formuladas al *chatbot*, se precisaría realizar estudios pertinentes acerca de técnicas de

Prompt Engineering y/o *fine-tuning*, puesto que a pesar de haber procurado realizar preguntas lo más acordes y coherentes posibles, no se han obtenido resultados óptimos.

5.2 Líneas futuras

Una vez concluidos los objetivos previstos del presente trabajo de fin de grado, se han podido observar durante la creación de este sistema de detección inteligente una serie de detalles que podrían continuar desarrollándose con el objetivo de seguir progresando en esta línea de investigación. Estos se podrían resumir en los siguientes puntos:

- Analizar los enlaces a páginas web que aparecen en los tuits publicados por diferentes usuarios puesto que, en muchos casos, aportarán información que pueda llegar a ser de utilidad a la hora de clasificar.
- Analizar las imágenes publicadas por los diferentes usuarios, pues al igual que los enlaces, pueden contener indicios que faciliten la clasificación del usuario en una de las clases.
- Como se ha podido comprobar, este proyecto ha centrado el foco en analizar los propios tuits publicados por el usuario rechazando los retuits. Sin embargo, se podría considerar como ampliación de este proyecto, hacer no solo un estudio sobre el nombre del usuario introducido sino ampliar este abanico a aquellos usuarios de los cuales haya retuiteado alguna publicación y determinar si estos pertenecen también al ámbito de defensa.
- Emplear y evaluar las salidas que se obtendrían mediante diferentes algoritmos de clasificación dentro de la rama de *Machine Learning* de aprendizaje automático supervisado alternativos a *Random Forest* y realizar una comparativa analizando las diversas métricas explicadas.
- Ampliar el campo de trabajo y realizar un estudio de cuentas con mayor influencia en este ámbito dentro de la red social, por ejemplo, mediante la API de *Search* u obteniendo un acceso para desarrolladores en la API de Twitter más elevado que permita implementar en el código funciones tales como: *search_users ()* en base a una palabra clave, *get_friends ()* para obtener los usuarios seguidos o *get_followers ()* para obtener sus seguidores. Comprobar de manera periódica si siguen cumpliendo con los requisitos pues los diferentes usuarios pueden cambiar sus intereses con el paso del tiempo.
- Repetir el experimento de clasificación de usuarios asistida por ChatGPT en sucesivas versiones del modelo (como ChatGPT-4).
- Creación de una interfaz para la herramienta como un servicio web el cual se pueda acceder a través de una URL en el navegador. En esta se introduciría el nombre del usuario a estudiar y se obtendría su clasificación.
- Pese que se ha procurado automatizar lo máximo posible la obtención de títulos para la creación del corpus necesario para el entrenamiento, es cierto que se precisaba de cierta supervisión para comprobar, entre otros que las subcategorías obtenidas continuasen perteneciendo o no al ámbito de estudio lo que ralentizaba el proceso en gran medida. Es por ello que, una de las posibles líneas futuras sería idear un método que automatice este proceso de creación del corpus en base a ciertos parámetros o instrucciones preestablecidos.

Cabe destacar, que este tipo de técnicas de clasificación podrían emplearse en otras redes sociales con el fin de obtener usuarios que publiquen de manera periódica y que resulten de interés en el ámbito de la defensa.

6 BIBLIOGRAFÍA

- [1] Y. Rodríguez, «GESI-Grupo de Estudios en Seguridad Internacional,» 6 Junio 2019. [En línea]. Available: Inteligencia de fuentes abiertas (OSINT): características, debilidades y engaño | GESI. [Último acceso: 15 Enero 2023].
- [2] Marlon Mike Toro Álvarez, Wilmer David Parada Jaimes, Martín Leonardo Bonilla-Duitama, «Research Gate,» Diciembre 2018. [En línea]. Available: https://www.researchgate.net/publication/329947099_Investigacion-del-Cibercrimen_y_Delitos-Informaticos_Utilizando_OSINT. [Último acceso: 14 Enero 2023].
- [3] J. C. Mejía-Llano, «Juan Carlos Mejía-Llano,» 9 Mayo 2022. [En línea]. Available: <https://www.juancmejia.com/marketing-digital/estadisticas-de-redes-sociales-usuarios-de-facebook-instagram-linkedin-twitter-whatsapp-y-otros-infografia/>. [Último acceso: 14 Enero 2023].
- [4] Fonte, Alberto, «Derecho de la red,» 8 Marzo 2021. [En línea]. Available: <https://derechodelared.com/osint/>. [Último acceso: 15 Enero 2023].
- [5] Web Empresa, «Web Empresa,» 1 Marzo 2018. [En línea]. Available: <https://www.webempresa.com/blog/que-es-twitter-como funciona-2.html>. [Último acceso: 15 Enero 2023].
- [6] L. Blanco, «Tíndalos,» 14 Julio 2022. [En línea]. Available: <https://tindalos.es/redes-sociales-mas-utilizadas-en-espana-en-2022/>. [Último acceso: 15 Enero 2023].
- [7] M. Ahlgreen, «Websiterating,» [En línea]. Available: <https://www.websiterating.com/es/research/twitter-statistics/>. [Último acceso: 15 Enero 2023].
- [8] Twitter, «Twitter Announces Second Quarter Results,» San Francisco, California, 2022.
- [9] S. Kemp, «DATAREPORTAL,» 15 Agosto 2022. [En línea]. Available: <https://datareportal.com/essential-twitter-stats>. [Último acceso: 22 Enero 2023].
- [10] J. Gutierrez, «Ciber Patrulla,» 22 Febrero 2022. [En línea]. Available: <https://ciberpatrulla.com/que-es-osint/>. [Último acceso: 17 Enero 2023].
- [] M. Frías, «Open Webinars,» 26 Julio 2021. [En línea]. Available: <https://openwebinars.net/blog/osint-que-es-tecnicas-y-herramientas/>. [Último acceso: 22 Enero 2023].

11] 2023].

[OSINTUX, «OSINTUX,» [En línea]. Available: <http://www.osintux.org/>. [Último acceso: 12] 25 Enero 2023].

[Kali , «Linux,» [En línea]. Available: <https://www.kali.org>. [Último acceso: 25 Enero 2023]. 13]

[«Parrot Os Security,» [En línea]. Available: <https://parrotlinux.org/>. [Último acceso: 25 14] Enero 2023].

[Sherlock Projects, «Sherlock Projects,» [En línea]. Available: <https://sherlock-project.github.io/>. [Último acceso: 25 Enero 2023]. 15]

[Teléfono Infoga, «Teléfono Infoga,» [En línea]. Available: 16] <https://sundowndev.github.io/phoneinfoga/>. [Último acceso: 25 Enero 2023].

[That'sTem-CI, «That'sTem-CI,» [En línea]. Available: 17] <https://thatsthem.com/challenge?r=%2F>. [Último acceso: 25 Enero 2023].

[Melissa, «Melissa,» [En línea]. Available: <https://www.melissa.com>. [Último acceso: 15 18] Enero 2023].

[DNS Lookup, «DNS Lookup,» [En línea]. Available: <https://dnslookup.es/>. [Último acceso: 19] 25 Enero 2023].

[IP Void, «IP Void,» [En línea]. Available: <https://www.ipvoid.com/>. [Último acceso: 25 20] Enero 2023].

[DNS Query, «DNS Query,» [En línea]. Available: <https://dnsquery.org/>. [Último acceso: 25 21] Enero 2023].

[DuckDuckGo, «DuckDuckGo,» [En línea]. Available: <https://duckduckgo.com/?va=b&t=hr>. 22] [Último acceso: 25 Enero 2023].

[MALTEGO, «MALTEGO,» [En línea]. Available: <https://www.maltego.com/>. [Último 23] acceso: 25 Enero 2023].

[Shodan, «Shodan,» [En línea]. Available: <https://www.shodan.io/>. [Último acceso: 25 Enero 24] 2023].

[Creepy, «Creepy,» [En línea]. Available: <https://www.geocreepy.com/>. [Último acceso: 25 25] Enero 2023].

[OSINT Framework, «OSINT Framework,» [En línea]. Available: 26] <https://osintframework.com/>. [Último acceso: 25 Enero 2023].

[Real Academia Española, «Real Academia Española,» [En línea]. Available: 27] <https://www.rae.es/observatorio-de-palabras/podcast>. [Último acceso: 26 Enero 2023].

[A. Naveira, «M4C Academy,» 8 Noviembre 2022. [En línea]. Available: 28] <https://marketing4ecommerce.net/historia-de-twitter/>. [Último acceso: 26 Enero 2023].

[M. Hughes, «TNW,» 1 Febrero 2018. [En línea]. Available: 29] <https://thenextweb.com/news/twitter-new-api-lets-developers-sift-every-tweet-history>. [Último acceso: 26 Enero 2023].

- [30] E. Castrejón, «Webadictos,» 3 Noviembre 2012. [En línea]. Available: <https://webadictos.com/breve-historia-de-twitter/>. [Último acceso: 27 Enero 2023].
- [31] R. Quintero, «GL-Grandes Logros,» 19 Mayo 2022. [En línea]. Available: <https://www.grandeslogros.co/historia-de-twitter/>. [Último acceso: 27 Enero 2023].
- [32] Busca Biografías, «Busca Biografías,» 17 Junio 2021. [En línea]. Available: <https://www.buscabiografias.com/biografia/verDetalle/11030>. [Último acceso: 25 Enero 2023].
- [33] Twitter, «Twitter API,» [En línea]. Available: <https://developer.twitter.com/en/docs/twitter-api>. [Último acceso: 20 Marzo 2023].
- [34] Twitter, «Centro de ayuda de Twitter,» 17 Enero 2023. [En línea]. Available: <https://help.twitter.com>. [Último acceso: 20 Febrero 2023].
- [35] Twitter, «Centro de ayuda,» [En línea]. Available: <https://help.twitter.com/es/using-twitter/follow-and-unfollow-topics#optout-interests>. [Último acceso: 28 Febrero 2023].
- [36] Twitter, «Twitter Empresas,» [En línea]. Available: <https://business.twitter.com/es/blog/twitter-analytics.html>. [Último acceso: 29 Febrero 2023].
- [37] Twitter Analytics, «Twitter Analytics,» [En línea]. Available: <https://analytics.twitter.com>. [Último acceso: 14 Marzo 2023].
- [38] Roialty, «Roialty,» 9 Noviembre 2016. [En línea]. Available: <https://roialty.com/what-is-a-network-and-why-are-interests-and-relationships-mapping-useful-for-the-brands/hashtag-network/>. [Último acceso: 25 Enero 2023].
- [39] Internet Archive, «Internet Archive,» 15 Octubre 2013. [En línea]. Available: <http://web.archive.org/>. [Último acceso: 24 Enero 2023].
- [40] TWINT, «TWINT,» 2013. [En línea]. Available: <https://www.twint.ch/en/>. [Último acceso: 29 Enero 2023].
- [41] KOLMITE, «KOLMITE,» 10 Diciembre 2022. [En línea]. Available: <https://kolmite.com/twint-herramienta-de-inteligencia-de-twitter/>. [Último acceso: 29 Enero 2023].
- [42] Zabrack, «GitHub,» 2022 Agosto 2022. [En línea]. Available: <https://github.com/Zabrack/TwoSINT>. [Último acceso: 29 Enero 2023].
- [43] Truth Nest, «Truth Nest,» [En línea]. Available: <https://www.truthnest.com/>. [Último acceso: 29 Enero 2023].
- [44] ODIN, «OSINT y Ciberinteligencia,» 5 Mayo 2022. [En línea]. Available: <https://odint.net/twitter-osint/>. [Último acceso: 29 Enero 2023].
- [45] TINFOLEAK, «TINFOLEAK,» [En línea]. Available: <https://tinfoleak.com/>. [Último acceso: 29 Enero 2023].
- [46] Vicente Aguilera Díaz, «Github,» 10 Abril 2018. [En línea]. Available: <https://github.com/vaguileradiaz/tinfoleak>. [Último acceso: 29 Febrero 2023].
- [47] All My Tweets, «All My Tweets,» [En línea]. Available: <https://www.allmytweets.net/connect/>. [Último acceso: 29 Enero 2023].
- [] Follower wonk, «Follower wonk,» [En línea]. Available: <https://followerwonk.com/>.

- 48] [Último acceso: 29 Enero 2023].
- [Impulso Creativo Publicidad, «Impulso Creativo,» 10 Octubre 2013. [En línea]. Available:
49] <https://www.impulsocreativo.es/blog/herramienta-para-analizar-twitter/>. [Último acceso: 29 Enero 2023].
- [Twitter Audit, «Twitter Audit,» 2012. [En línea]. Available: <https://twitteraudit.com/>.
50] [Último acceso: 29 Enero 2023].
- [Abbas Alidina, «Crowdbabble,» 18 Junio 2018. [En línea]. Available:
51] <https://www.crowdbabble.com/blog/the-ultimate-twitter-audit-guide/>. [Último acceso: 25 Enero 2023].
- [Grupo de investigación WebTlab, «T-hoarder,» [En línea]. Available:
52] <http://thoarder.ddns.net>. [Último acceso: 29 Febrero 2023].
- [Julio Omar Ancízar Palacio Niño, «Detección de comunidades en redes: Algoritmos y
53] aplicaciones,» 2013. [En línea]. Available: <https://digibug.ugr.es/handle/10481/55384>. [Último acceso: 29 Febrero 2023].
- [Liskarys Rojas, «Nido Colectivo,» 10 Septiembre 2021. [En línea]. Available:
54] <https://nidocolectivo.com/blog/que-son-las-comunidades-de-twitter>. [Último acceso: 30 Enero 2023].
- [Najera, Christian Iván Ledesma Bermúdez y Abel García, *Detección de comunidades en
55] redes sociales por medio de un algoritmo de agrupación dinámico en alta definición. Posgrado en Ciencias Naturales e Ingeniería*, Mexico, 2018.
- [Mateo, Adrián Lorenzo, *Detección de comunidades en redes sociales, estudio sobre las
56] elecciones catalanas y elecciones generales de 2015. UAM, Grado en Ingeniería Informática*, Madrid, 2016.
- [Jeff Hawkins, Sandra Blakeslee, *On Intelligence*, 2004.
57]
- [Factor Humano, «Factor Humano,» 28 Febrero 2007. [En línea]. Available:
58] <https://factorhumano.wordpress.com/2007/02/28/on-intelligence-jeff-hawkins/>. [Último acceso: 5 Marzo 2023].
- [SAS, «SAS,» [En línea]. Available: https://www.sas.com/es_es/insights/analytics/what-is-artificial-intelligence.html. [Último acceso: 6 Marzo 2023].
59]
- [Inloc Robotics, «Inloc Robotics,» 3 Marzo 2021. [En línea]. Available:
60] <https://inlocrobotics.com/es/evolucion-de-la-inteligencia-artificial/>. [Último acceso: 6 Marzo 2023].
- [NetApp, «NetApp,» [En línea]. Available: <https://www.netapp.com/es/artificial-intelligence/what-is-artificial-intelligence/>. [Último acceso: 6 Marzo 2023].
61]
- [Peter Norvig y Stuart Rusell, *Artificial Intelligence: A Modern Approach*, 1995.
62]
- [Peñarroya, Montserrat, «Montse Peñarroya,» 2 Noviembre 2020. [En línea]. Available:
63] <https://www.montsepenarroya.com/campos-en-los-que-se-aplica-la-inteligencia-artificial-y-nuevas-profesiones-no-cientificas/>. [Último acceso: 6 Marzo 2023].

- [Iberdrola, «Qué es el 'Machine Learning',» [En línea]. Available:
64] <https://www.iberdrola.com/innovacion/machine-learning-aprendizaje-automatico>. [Último
acceso: 2023 Marzo 8].
- [Repsol, «Machine Learning,» [En línea]. Available: [https://www.repsol.com/es/energia-](https://www.repsol.com/es/energia-futuro/tecnologia-innovacion/machine-learning)
65] [futuro/tecnologia-innovacion/machine-learning](https://www.repsol.com/es/energia-futuro/tecnologia-innovacion/machine-learning). [Último acceso: 5 Marzo 2023].
- [PowerData, «Big Data,» 2022, [En línea]. Available: <https://www.powerdata.es/big-data>.
66] [Último acceso: 5 Marzo 2023].
- [Oracle, «IoT,» [En línea]. Available: [https://www.oracle.com/es/internet-of-things/what-is-](https://www.oracle.com/es/internet-of-things/what-is-iot/)
67] [iot/](https://www.oracle.com/es/internet-of-things/what-is-iot/). [Último acceso: 5 Marzo 2023].
- [TIBCO, «Machine Learning,» [En línea]. Available: [https://www.tibco.com/es/reference-](https://www.tibco.com/es/reference-center/what-is-supervised-learning)
68] [center/what-is-supervised-learning](https://www.tibco.com/es/reference-center/what-is-supervised-learning). [Último acceso: 6 Marzo 2023].
- [DataScientest, «Random Forest,» 25 Enero 2023. [En línea]. Available:
69] <https://datascientest.com/es/random-forest-bosque-aleatorio-definicion-y-funcionamiento>.
[Último acceso: 7 Marzo 2023].
- [González, Ligdi, «AprendeIA Random Forest,» 23 Marzo 2018. [En línea]. Available:
70] <https://aprendeia.com/aprendizaje-supervisado-random-forest-classification/>. [Último acceso: 6
Marzo 2023].
- [Heras, José Martínez, «IArtificial,» 18 Septiembre 2020. [En línea]. Available:
71] <https://www.iartificial.net/random-forest-bosque-aleatorio/>. [Último acceso: 7 Marzo 2023].
- [Wood, Thomas, «DeepAI,» [En línea]. Available: [https://deepai.org/machine-learning-](https://deepai.org/machine-learning-glossary-and-terms/random-forest)
72] [glossary-and-terms/random-forest](https://deepai.org/machine-learning-glossary-and-terms/random-forest). [Último acceso: 7 Marzo 2023].
- [IBM, «Aprendizaje supervisado,» [En línea]. Available: [https://www.ibm.com/es-](https://www.ibm.com/es-es/topics/supervised-learning)
73] [es/topics/supervised-learning](https://www.ibm.com/es-es/topics/supervised-learning). [Último acceso: 7 Marzo 2023].
- [MathWorks, «MathWorks,» [En línea]. Available:
74] <https://es.mathworks.com/discovery/support-vector-machine.html>. [Último acceso: 8 Marzo
2023].
- [Jairo, «Las cosas de Internet,» 10 Diciembre 2022. [En línea]. Available:
75] <https://lascosasdeinternet.com/tech-development/que-son-los-modelos-de-lenguaje-llm-en-ia/>.
[Último acceso: 14 Marzo 2023].
- [OpenAI, «ChatGPT,» [En línea]. Available: <https://chat.openai.com/chat>. [Último acceso:
76] 14 Marzo 2023].
- [Romero, Marta Sanz, «Computer Hoy,» 1 Agosto 2020. [En línea]. Available:
77] <https://computerhoy.com/reportajes/industria/gpt-3-que-es-687371>. [Último acceso: 14 Marzo
2023].
- [OpenAI, «API OpenAI,» [En línea]. Available: <https://openai.com/blog/openai-api>. [Último
78] acceso: 10 Marzo 2023].
- [Planeta Chatbot, «Planeta Chatbot,» 21 Agosto 2019. [En línea]. Available:
79] <https://planetachatbot.com/linea-tiempo-historia-de-chatbots-antes-ahora-y-manana>. [Último
acceso: 14 Marzo 2023].
- [Le Grand Continent, «Le Grand Continent,» 16 Marzo 2023. [En línea]. Available:
<https://legrandcontinent.eu/es/2023/03/16/de-que-es-capaz-gpt-4>. [Último acceso: 17 Marzo

- 80] 2023].
- [OpenAI, «GPT-4,» 14 Marzo 2023. [En línea]. Available: <https://openai.com/product/gpt-4>.
81] [Último acceso: 14 Marzo 2023].
- [M. Merino, «GENBETA,» 14 Marzo 2023. [En línea]. Available:
82] <https://www.genbeta.com/actualidad/esta-aqui-gpt-4-nueva-inteligencia-artificial-openai-que-nos-permitira-hablar-chatgpt-usando-imagenes>. [Último acceso: 14 Marzo 2023].
- [Á. H. Ruiz, «Herramientas de OSINT para la obtención de información en Twitter sobre el
83] conflicto Ucrania-Rusia. TFG CUD-ENM,» Pontevedra, 2021.
- [VirtualBox, «VirtualBox,» 17 Enero 2023. [En línea]. Available:
84] <https://www.virtualbox.org/>. [Último acceso: 10 Febrero 2023].
- [Python, «Python,» [En línea]. Available: <https://www.python.org/downloads/>. [Último
85] acceso: 20 Febrero 2023].
- [TIOBE, «TIOBE,» Marzo 2023. [En línea]. Available: <https://www.tiobe.com/tiobe-index/>.
86] [Último acceso: 2 Marzo 2023].
- [Sublime Text, «Sublime Text,» [En línea]. Available: <https://www.sublimetext.com/3>.
87] [Último acceso: 22 Febrero 2023].
- [Hugging Face, «Datasets Wikipedia,» [En línea]. Available:
88] <https://huggingface.co/datasets/wikipedia>. [Último acceso: 1 Marzo 2023].
- [Hugging Face, «Datasets index,» [En línea]. Available:
89] <https://huggingface.co/docs/datasets/index>. [Último acceso: 5 Marzo 2023].
- [JSON, «JSON,» [En línea]. Available: <https://www.json.org/json-en.html>. [Último acceso:
90] 26 Febrero 2023].
- [Vishal, «PYnativo,» 5 Enero 2023. [En línea]. Available: <https://pynative.com/python-save-dictionary-to-file/#h-how-to-save-a-dictionary-to-file-in-python>. [Último acceso: 28 Febrero
91] 2023].
- [MediaWiki, «MediaWiki,» 30 Noviembre 2022. [En línea]. Available:
92] https://www.mediawiki.org/wiki/API:Main_page/es. [Último acceso: 2 Marzo 2023].
- [IPTC, «IPTC. Information Technology For News,» 15 Octubre 2010. [En línea]. Available:
93] <https://cv.iptc.org/newscodes/subjectcode/>. [Último acceso: 28 Febrero 2023].
- [Wikipedia, «Categories,» 12 Abril 2021. [En línea]. Available:
94] <https://en.wikipedia.org/wiki/Wikipedia:Contents/Categories>. [Último acceso: 28 Febrero 2023].
- [Wikipedia, «National security,» 29 Noviembre 2022. [En línea]. Available:
95] https://en.wikipedia.org/wiki/Category:National_security. [Último acceso: 28 Marzo 2023].
- [Wikipedia, «Espacio de nombres,» 2 Junio 2021. [En línea]. Available:
96] https://es.wikipedia.org/wiki/Ayuda:Espacio_de_nombres. [Último acceso: Febrero 2023].
- [J. Roesslein, «Tweepy,» 2009. [En línea]. Available:
97] <https://docs.tweepy.org/en/latest/index.html>. [Último acceso: 26 Febrero 2023].
- [Wurster, Taehoon Kim y Kevin, «Pypi,» 31 Octubre 2022. [En línea]. Available:

- 98] <https://pypi.org/project/emoji/>. [Último acceso: 7 Marzo 2023].
- [Python, «LangDetect,» 7 Mayo 2021. [En línea]. Available:
- 99] <https://pypi.org/project/langdetect/>. [Último acceso: 8 Marzo 2023].
- [Python, «Googletrans,» 14 Junio 2020. [En línea]. Available:
- 100] <https://pypi.org/project/googletrans/>. [Último acceso: 7 Marzo 2023].
- [Python, «Librería 're',» 3 Marzo 2023. [En línea]. Available:
- 101] <https://docs.python.org/es/3/library/re.html>. [Último acceso: 7 Marzo 2023].
- [Unipython, «Unipython,» [En línea]. Available: [https://unipython.com/el-modelo-](https://unipython.com/el-modelo-embeddings-incrustaciones-de-palabras/)
- 102] [embeddings-incrustaciones-de-palabras/](https://unipython.com/el-modelo-embeddings-incrustaciones-de-palabras/). [Último acceso: 10 Marzo 2023].
- [Pandas, «Pandas,» 31 Agosto 2022. [En línea]. Available: <https://pandas.pydata.org/>.
- 103] [Último acceso: 10 Marzo 2023].
- [OpenAI, «Librerías,» [En línea]. Available: <https://platform.openai.com/docs/libraries>.
- 104] [Último acceso: 10 Marzo 2023].
- [Hugging Face, «Transformers,» [En línea]. Available:
- 105] <https://huggingface.co/docs/transformers/v4.19.0/es/index>. [Último acceso: 10 Marzo 2023].
- [Plotly, «Plotly,» [En línea]. Available: <https://plotly.com/python/>. [Último acceso: 10 Marzo
- 106] 2023].
- [Matplotlib, «Matplotlib,» [En línea]. Available: <https://matplotlib.org/>. [Último acceso: 10
- 107] Marzo 2023].
- [González, Ligdi, «Aprende IA-Librería Scikit Learn,» [En línea]. Available:
- 108] <https://aprendeia.com/libreria-scikit-learn-de-python/>. [Último acceso: 10 Marzo 2023].
- [PyTorch, «PyTorch,» [En línea]. Available: <https://pytorch.org/vision/stable/index.html>.
- 109] [Último acceso: 10 Marzo 2023].
- [Unipython, «Unipython,» [En línea]. Available: [https://unipython.com/scipy-funciones-](https://unipython.com/scipy-funciones-principales/)
- 110] [principales/](https://unipython.com/scipy-funciones-principales/). [Último acceso: 10 Marzo 2023].
- [GitHub, «OpenAI,» 10 Enero 2023. [En línea]. Available: [https://github.com/openai/openai-](https://github.com/openai/openai-cookbook/blob/main/examples/Obtain_dataset.ipynb)
- 111] [cookbook/blob/main/examples/Obtain_dataset.ipynb](https://github.com/openai/openai-cookbook/blob/main/examples/Obtain_dataset.ipynb). [Último acceso: 10 Marzo 2023].
- [GitHub, «Classification using embeddings,» 10 Junio 2022. [En línea]. Available:
- 112] [https://github.com/openai/openai-](https://github.com/openai/openai-cookbook/blob/main/examples/Classification_using_embeddings.ipynb)
[cookbook/blob/main/examples/Classification_using_embeddings.ipynb](https://github.com/openai/openai-cookbook/blob/main/examples/Classification_using_embeddings.ipynb). [Último acceso: 10
- Marzo 2023].
- [Sitio Big Data, «Sitio Big Data,» 19 Enero 2019. [En línea]. Available:
- 113] <https://sitiobigdata.com/2019/01/19/machine-learning-metrica-clasificacion-parte-3>. [Último
- acceso: 16 Marzo 2023].
- [L. Ramírez, «IEB School,» 2 Marzo 2023. [En línea]. Available:
- 114] <https://www.iebschool.com/blog/prompt-engineering-tecnologia/>. [Último acceso: 17 Marzo
- 2023].

ANEXO I: IMPLICACIONES SOCIALES, ECONÓMICAS Y AMBIENTALES

El desarrollo y uso de la tecnología implica la responsabilidad y la consideración de las consecuencias sociales y ambientales que esta puede traer consigo y tomar las pertinentes medidas para mitigar posibles riesgos o impactos negativos y no deseados.

Si bien es cierto que, a pesar de no ser las implicaciones ambientales las más destacables de este proyecto cabe mencionar ciertos aspectos que tienen repercusiones en el medio ambiente. En primer lugar, el desarrollo de dicho sistema de detección inteligente (así como la realización del enfrentamiento del modelo con ChatGPT) requiere de un gran consumo de energía, pues son muchas las horas que se han precisado para operar y procesar cantidades ingentes de datos, así como para la realización de la presente memoria, aumentando todo ello la huella de carbono en el ecosistema. Además, este gran consumo de energía puede tener indirectas implicaciones en el cambio climático debido a las emisiones de gases de efecto invernadero que se desprenden. A su vez, hay que tener en cuenta todos los recursos computacionales empleados puesto todo ello supone tanto la explotación de recursos naturales para su creación como un futuro desecho electrónico cuando su ciclo de vida finalice.

Por último, se ha llevado a cabo un estudio económico sobre el coste que tendría el desarrollo de dicho sistema de detección inteligente de usuarios interesados en el ámbito de la defensa. Para ello se han ido contabilizando las horas invertidas necesarias para el desarrollo de cada una de las partes del proyecto, desde sus inicios con la introducción del trabajo, la contextualización realizada en el epígrafe del estado del arte y el desarrollo del código hasta los resultados obtenidos y las conclusiones a las cuales se ha llegado una vez finalizada y evaluada la herramienta. Es decir, se han tenido en cuenta tanto las horas previas de investigación y profundización en el ámbito en cuestión, así como las horas invertidas en el desarrollo del código y la presente memoria. En la siguiente Tabla A1-1 muestra de forma sintetizada la cuantía de horas invertidas desglosadas epígrafe por epígrafe.

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	H.TOT.
1. INTRO. Y OBJETIVOS	6									6		12
1.1 INTRO. Y MOTIVACIÓN	5									2,5		
1.2 OBJETIVOS Y METODOLOGÍA	1									1,5		
1.3 ESTRUCTURA	0									2		
2. ESTADO DEL ARTE	4,5	2,5	9,5	12,5	5	3			11,5			48,5
2.1 OSINT	4,5	2,5	4,5									
2.2 TWITTER			5	0,5								
2.3 OSINT Y TWITTER				7,5	1,5	3						
2.4 COMUNIDADES EN TWITTER				4,5	2,5							
2.5 INTELIGENCIA ARTIFICIAL									10,5			
2.6 TRABAJOS RELACIONADOS					1				1			
3. DESARROLLO							7,5	9,5	11,5	9		37,5
3.1 METODOLOGÍA							1					
3.2 WIKIPEDIA							6,5	2				
3.3 CORPUS ARTÍCULOS								6,5				
3.4 OBTENCIÓN TUIITS									4			
3.5 CLASIFICADOR								1	7,5	9		
4. RESULTADOS							0,5			8,5		9
							0,5			8,5		
5. CONCLUSIONES										4		4
										4		
ANEXOS							1		3	4		8
							1		3	4		
FORMATO		2,5	3,5				0,5		1	1		8,5

			2,5	3,5			0,5		1	1			
CORRECCIONES	9,5											7	16,5
			9,5									7	
PROGRAMACIÓN INICIOS	12												12
Instalación, iniciación...					12								
CÓDIGO FINAL	13 16,5 14,5 28 10												82
						13	16,5	14,5	28	10			
													238

Tabla A1-1 Horas totales invertidas en el desarrollo del proyecto. (Autoría propia)

Cabe destacar, que para la realización de este resumen final se han ido anotando día a día las horas empleadas especificando en qué epígrafe en concreto en las tablas mostradas a continuación (véanse Tabla A1-2, Tabla A1-3 y Tabla A1-4):

	SEMANA 1 (9ene-15ene)					SEMANA 2 (16ene-22ene) (CISI)							SEMANA 3 (23ene-29ene)						
	X-11	J-12	V-13	S-14	D-15	L-16	M-17	X-18	J-19	V-20	S-21	D-22	L-23	M-24	X-25	J-26	V-27	S-28	D-29
1. INTRO. Y OBJETIVOS																			
1.1 INTRO. Y MOTIVACIÓN	4	1																	
1.2 OBJETIVOS Y METODOLOGÍA		1																	
1.3 ESTRUCTURA																			
2. ESTADO DEL ARTE																			
2.1 OSINT		1	1,5	1	1					1,5	0,5	0,5	1	1		2	0,5		
2.2 TWITTER																	1	1	3
2.3 OSINT Y TWITTER																			
2.4 COMUNIDADES EN TWITTER																			
2.5 INTELIGENCIA ARTIFICIAL																			
2.6 TRABAJOS RELACIONADOS																			
3. DESARROLLO																			
3.1 METODOLOGÍA																			
3.2 WIKIPEDIA																			
3.3 CORPUS ARTÍCULOS																			
3.4 OBTENCIÓN TUIITS																			
3.5 CLASIFICADOR																			
4. RESULTADOS																			
5. CONCLUSIONES																			
ANEXOS																			
FORMATO																			
CORRECCIONES																			
PROGRAMACIÓN INICIOS																			
Instalación, iniciación...																			
CÓDIGO FINAL																			

Tabla A1-2 Horas semanales invertidas en el desarrollo del proyecto I. (Autoría propia)

SISTEMA DE DETECCIÓN INTELIGENTE DE USUARIOS DE TWITTER INTERESADOS
EN LA TEMÁTICA DE DEFENSA

SEMANA 4 (30ene-5feb)							SEMANA 5 (6feb-12feb)							SEMANA 6 (13feb-19feb)							SEMANA 7 (20feb-26feb)						
L-30	M-31	X-1	J-2	V-3	S-4	D-5	L-6	M-7	X-8	J-9	V-10	S-11	D-12	L-13	M-14	X-15	J-16	V-17	S-18	D-19	L-20	M-21	X-22	J-23	V-24	S-25	D-26
0,5 3,5 2 1 1 1 0,5 1 1							1 0,5 1,5 1 1							1 2													
																					1 1 2 1 2,5						
																					0,5						
																					1						
1 1 1,5																											
1 1,5 4 2 1																											
							1,5 3 2 2 1,5 2																				
														2 3 2 3 3							2 5 2 2 2,5 2 1						

Tabla A1-3 Horas semanales invertidas en el desarrollo del proyecto II. (Autoría propia)

SEMANA 8 (27feb-5mar)							SEMANA 9 (6mar-12mar)							SEMANA 10 (13mar-19mar)							SEMANA 11 (20mar-24mar)				
L-27	M-28	X-1	J-2	V-3	S-4	D-5	L-6	M-7	X-8	J-9	V-10	S-11	D-12	L-13	M-14	X-15	J-16	V-17	S-18	D-19	L-20	M-21	X-22	J-23	V-24
																			2,5						
																			1,5						
																			2						
											1,5	5	4												
												1													
2																									
	1,5	2	1	2			2	1	1																
							1	1	0,5	2	2	1		2	3	4									
																0,5	2,5	1,5	2		2				
																			3		1				
														3				3			1				
														1	1										
																						2	3	2	

4 3 3 2,5 1 1 | 4 5 5 4 5 3 2 | 2 3 2 3

Tabla A1-4 Horas semanales invertidas en el desarrollo del proyecto III. (Autoría propia)

Se requieren pues, un total de 238 horas de mano de obra para el desarrollo de la herramienta en cuestión. Tomando como referencia un precio promedio de mano de obra de 20 €/hora para el desarrollo del código y 18 €/hora para la realización de memoria. Se obtiene un total de 4.472 € en mano de obra (véase desglose de precios en la Tabla A1-5 a continuación).

Programación (horas)	Precio programación (€/hora)	€
94	20	1880
Memoria (horas)	Precio desarrollo memoria (€/hora)	€
144	18	2592

Total M.O. (€)	4472
----------------	------

Tabla A1-5 Desglose del coste total de mano de obra. (Autoría propia)

Habr  que tener en cuenta a su vez, el coste adicional que tiene el empleo de la API de OpenAI en funci n de las horas empleadas y las actividades realizadas con ella. En este caso, su uso ha radicado en la extracci n de los diferentes conjuntos de *embeddings* para tanto el entrenamiento como la evaluaci n del clasificador, as  como las diferentes peticiones para emplear ChatGPT. A continuaci n, se procede a mostrar una captura del coste de este servicio (v ase Figura A1-1).

Usage



Figura A1-1 Coste derivado del empleo de la API de OpenAI. (Autor a propia)

Se puede observar un coste total de 11,68 \$ como resultado de emplear dicha interfaz para desarrolladores, lo cual equivale a 10,72 €. Esto implica que el coste total para llevar a cabo el proyecto de desarrollo de esta herramienta serían 4.482,72 €.

ANEXO II: REFLEXIONES ÉTICAS Y SOCIALES

El desarrollo de un sistema de detección inteligente de usuarios de la red social Twitter relacionados o de interés para el ámbito de la defensa plantea diversas cuestiones éticas y sociales que deben ser abordadas. En general, el tratamiento de datos extraídos de plataformas sociales puede acarrear grandes repercusiones de esta entidad, especialmente si se llevan a cabo sin la expresa autorización o consideración de los derechos y privacidad de cada uno de los usuarios.

Las redes sociales son una fuente incesante de información, a partir de la cual se pueden obtener datos personales además de datos de comportamiento de los diferentes usuarios, entre otros. Todo ello se ha convertido en un poderoso conjunto de información que ha derivado en una incesante lucha entre diferentes compañías que buscan conseguir su control para, mediante diferentes análisis y algoritmos de entre otros, poder detectar conductas de comportamiento, sentimientos de usuarios, tendencias, etc., de las cuales poder beneficiarse y tomar decisiones mediante la creación de estrategias empresariales.

El significativo crecimiento e importancia que las redes sociales están adquiriendo en el día a día de las personas, ha derivado en diferentes estudios e investigaciones que plantean la necesidad de implantar medidas a nivel internacional en pos de la seguridad en el tratamiento de la información privada de cada usuario. Algunas de estas medidas incluyen:

- La regulación de la privacidad de los datos. Muchos son los países que han aprobado leyes que velan por la seguridad y privacidad de los datos de los usuarios. La Unión Europea, entre otros, ha establecido unas estrictas normas para el procesamiento y la transferencia de información personal otorgando a los propios clientes el derecho de controlar sus datos personales en el Reglamento General de Protección de Datos (RGPD).
- Transparencia en la publicidad política: Durante los últimos años con el apogeo de las redes sociales, ha quedado demostrado el empleo de estas plataformas como medio de desinformación tanto en elecciones como en campañas políticas. Como consecuencia, se han establecido normativas que velan por la transparencia en la publicidad política. Cabe destacar el hecho de que la red social Facebook ha implantado un sistema de verificación de anuncios en el ámbito de la política en varios países de la geografía mundial con el fin de garantizar la identificación de la entidad financiadora de dicha publicidad.
- Lucha contra la desinformación y el discurso de odio. Cada vez son más los países que han optado por establecer restricciones y penalizar a usuarios que realicen publicaciones con contenido de odio, así como para combatir la desinformación en todos los campos.
- Responsabilidad de las plataformas tales como Facebook, Instagram, Twitter, etc. Estas cada vez sufren una mayor presión para asumir responsabilidad por el contenido que suben los diferentes usuarios a sus plataformas. Se han presentado muchas demandas contra este tipo de empresas por contenido perjudicial y se les exige regulaciones más estrictas para evitarlo.

En primer lugar, cabe señalar que, durante toda la ejecución del vigente proyecto, la extracción de todos los datos de la plataforma de Twitter se ha realizado a través de su propia API de desarrolladores con el consentimiento pertinente, y no accediendo bajo ningún concepto a información clasificada como privada en dicha red social. Es decir, todos los datos recopilados y empleados para la consecución de esta herramienta son públicos. Todo esto es aplicable al conjunto de datos extraídos de la página web de Wikipedia obtenidos para desarrollar el clasificador.

Uno de los aspectos más importantes en cuanto al tratamiento de datos de los usuarios en las diferentes redes sociales se refiere, es la transparencia que ha de tener el proceso completo así como el fin último de la obtención de estos. Se ha de informar y dejar claramente plasmado el método de recopilación, almacenamiento y empleo final de los mismos garantizando, de este modo, la privacidad y seguridad de toda la información extraída.

Es por ello por lo que, una de las misiones fundamentales que debe desarrollar la propia API es, mediante un equipo de expertos en ética y seguridad informática y de la información, la investigación cada una de las personas que solicitan altos permisos de desarrolladores para cerciorarse de mitigar cualquier riesgo que pudiera desencadenar el empleo de la información, accesible con dichas credenciales, con fines malintencionados.

No obstante, hay que ser precavidos también con el empleo que pueden dar ciertas personas al tipo de herramientas como la desarrollada en el este proyecto de fin de grado, pues puede hacerse un uso indebido de las mismas para monitorear a ciertos usuarios, detectados mediante dichos sistemas inteligentes, por meras razones políticas o personales en vez de simplemente por motivos legítimos de seguridad.

En conclusión, el desarrollo de dicha herramienta de detección inteligente que emplea información de la red social Twitter, plantea desafíos tanto éticos como sociales que deben ser tenidos en cuenta, pues es de vital importancia, garantizar la privacidad y seguridad de cada uno de los usuarios. Es necesaria a su vez, la supervisión por personal especializado en estas cuestiones para no causar ningún daño verificando que la recopilación y el empleo de toda la información se hace con fines beneficiosos para la sociedad en general.