

Universidades de Burgos, León y
Valladolid

Máster universitario

Inteligencia de Negocio y Big Data en Entornos Seguros



TFM del Máster Inteligencia de Negocio
y Big Data en Entornos Seguros

Herramienta *open-source* para
análisis de sentimientos en redes
sociales sobre palabras clave o
temas específicos

Presentado por Liviu Viorel Jula Vacar
en Universidad de Burgos — 1 de marzo
de 2023

Tutor: Dr. Álgvar Arnaiz González

Universidades de Burgos, León y Valladolid



Máster universitario en Inteligencia de Negocio y Big Data en Entornos Seguros

D. Álgar Arnaiz González, profesor del departamento de Ingeniería Informática, Área de Lenguajes y Sistemas Informáticos.

Expone:

Que el alumno D. Liviu Viorel Jula Vacar, con DNI dni, ha realizado el Trabajo final de Máster en Inteligencia de Negocio y Big Data en Entornos Seguros titulado “Herramienta *open-source* para análisis de sentimientos en redes sociales sobre palabras clave o temas específicos”.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 1 de marzo de 2023

Vº. Bº. del Tutor:

D. Álgar Arnaiz González

Resumen

Las opiniones públicas que se realizan en Internet sobre diversos productos, servicios, marcas o lugares pueden influenciar el comportamiento de las personas. La gran mayoría de estas opiniones se forman en redes sociales, foros de discusión y en los diferentes sitios web de reseñas.

El objetivo de este trabajo será emplear la información disponible públicamente para crear una herramienta que permita realizar un análisis de sentimientos sobre ciertas palabras clave o temas específicos de los que se quiera obtener información. Se realizará un proceso ETL para el procesamiento de los datos y se implementará un *dashboard* que permita visualizar y explorar la información obtenida finalmente.

Para conseguir estos objetivos, se emplearán tecnologías *open-source* para el desarrollo de la herramienta y técnicas de procesamiento de lenguaje natural para el análisis de sentimientos basado en aspectos clave.

Descriptores

Aprendizaje automático, procesamiento de lenguaje natural, big data, ETL, *dashboard*, visualización de datos, *open-source*.

Abstract

Public opinions made on the Internet about various products, services, brands or places can influence people's behavior. The vast majority of these opinions are formed on social media, discussion forums and on the different review websites.

The objective of this project will be to make use of publicly available information to create a tool able to perform sentiment analysis on certain keywords or specific topics for which we want to obtain information. An ETL workflow will be used to process the data and a dashboard will be implemented to visualize and explore the final information obtained.

To achieve these objectives, open-source technologies will be used for the development of the tool and natural language processing techniques for aspect-based sentiment analysis.

Keywords

Machine learning, natural language processing, big data, ETL, dashboard, data visualization, open-source.

Índice general

Índice general	iii
Índice de figuras	v
Índice de tablas	vi
1. Introducción	3
Memoria	3
2. Objetivos del proyecto	5
3. Conceptos teóricos	7
4. Técnicas y herramientas	9
4.1. Técnicas	9
4.2. Herramientas	10
5. Aspectos relevantes del desarrollo del proyecto	15
5.1. Extracción de datos	15
6. Trabajos relacionados	21
6.1. Herramientas similares	21
6.2. Artículos científicos	23
7. Conclusiones y Líneas de trabajo futuras	25

Apéndices	26
Apéndice A Plan de Proyecto Software	29
A.1. Introducción	29
A.2. Planificación temporal	29
A.3. Estudio de viabilidad	33
Apéndice B Especificación de Requisitos	35
B.1. Introducción	35
B.2. Objetivos generales	35
B.3. Catalogo de requisitos	35
B.4. Especificación de requisitos	35
Apéndice C Especificación de diseño	37
C.1. Introducción	37
C.2. Diseño de datos	37
C.3. Diseño procedimental	37
C.4. Diseño arquitectónico	37
Apéndice D Documentación técnica de programación	39
D.1. Introducción	39
D.2. Estructura de directorios	39
D.3. Manual del programador	39
D.4. Compilación, instalación y ejecución del proyecto	39
D.5. Pruebas del sistema	39
Apéndice E Documentación de usuario	41
E.1. Introducción	41
E.2. Requisitos de usuarios	41
E.3. Instalación	41
E.4. Manual del usuario	41
Bibliografía	43

Índice de figuras

A.1. Gráfico <i>burn-down</i> del <i>Sprint 0</i>	31
---	----

Índice de tablas

5.1. Recursos disponibles a través de la <i>API</i> de Twitter	16
5.2. Muestra de nodos disponibles en <i>Graph API</i> de Facebook . . .	17
5.3. Muestra de nodos disponibles en <i>Graph API</i> de Instagram . . .	18
5.4. Recursos disponibles a través de la <i>API</i> de YouTube	18
5.5. Recursos disponibles a través de la <i>API</i> de Reddit	19

Memoria

Introducción

Descripción del contenido del trabajo y del estructura de la memoria y del resto de materiales entregados.

Objetivos del proyecto

Este apartado explica de forma precisa y concisa cuales son los objetivos que se persiguen con la realización del proyecto. Se puede distinguir entre los objetivos marcados por los requisitos del software a construir y los objetivos de carácter técnico que plantea a la hora de llevar a la práctica el proyecto.

Conceptos teóricos

En aquellos proyectos que necesiten para su comprensión y desarrollo de unos conceptos teóricos de una determinada materia o de un determinado dominio de conocimiento, debe existir un apartado que sintetice dichos conceptos.

Técnicas y herramientas

Esta parte de la memoria tiene como objetivo presentar las técnicas metodológicas y las herramientas de desarrollo que se han utilizado para llevar a cabo el proyecto. En el caso de algunas de estas herramientas se estudiarán diferentes alternativas, en las que se incluirán comparativas entre las distintas opciones y una justificación de la elección realizadas.

4.1. Técnicas

En este apartado se hará una breve descripción sobre las técnicas empleadas a lo largo del proyecto.

SCRUM

Es un proceso de desarrollo software enfocado hacia las metodologías ágiles. Consiste en segmentar un proyecto en varios requisitos que se han de cumplir y posteriormente subdividir estos en tareas. El desarrollo se realiza mediante *sprints*, iteraciones incrementales de normalmente dos semanas de duración, en los que se planifican las tareas a realizar durante dicho periodo.

Procesamiento de Lenguaje Natural

El término *NLP* (*Natural Language Processing*) se refiere al conjunto de métodos dentro de la inteligencia artificial que trabajan con recursos textuales o sonoros. Se ponen en práctica metodologías de estadística, lingüística y *machine learning* para permitir crear programas que puedan interpretar dicho tipo de información.

Sentiment Analysis

El análisis de sentimientos es una técnica en la que se busca identificar y extraer información subjetiva a partir de recursos textuales. Las principales maneras de realizar este tipo de análisis siguen dos rutas.

La primera, utilizando reglas y diccionarios de palabras a las que se les asigna distintas puntuaciones según el sentimiento asociado a cada palabra. La segunda, y la que mejores resultados proporciona actualmente, emplea técnicas de *NLP* para extraer características de los datos y comprender el contexto de la información proporcionada. Esto permite realizar clasificaciones y predicciones más acertadas ya que el resultado no se limita simplemente a un subconjunto de palabras, sino al sentido que se les da a las mismas también.

4.2. Herramientas

Para llevar a cabo este proyecto, se ha utilizado el siguiente conjunto de herramientas.

GitHub

Para el *hosting* del repositorio se ha utilizado *GitHub*¹, puesto que ya se tenía experiencia en el uso de esta plataforma. Permite realizar la gestión del control de versiones a lo largo del desarrollo del software y simplifica el seguimiento de las tareas. Posee capacidades para creación de procesos de integración continua y despliegue continuo (*CI/CD*), automatización de flujos de trabajo, seguimiento y gestión de proyectos.

ZenHub

Para facilitar el trabajo de la gestión del proyecto se ha utilizado *ZenHub*². Es una plataforma centrada en mejorar la productividad de los equipos de desarrollo, que permite llevar a cabo la planificación del proyecto, realizar un seguimiento del progreso y calcular métricas de productividad mediante gráficas.

Se ha elegido esta herramienta ya que, además de permitir realizar toda la gestión del proyecto, cuenta con una extensión web desde la que se puede

¹<https://github.com/>

²<https://www.zenhub.com/>

acceder al panel de control directamente desde el propio repositorio de GitHub. Por lo que todas las operaciones de planificación de tareas se llevan a cabo desde el mismo lugar y facilita el trabajo del desarrollador.

Entorno de desarrollo integrado (IDE)

Herramientas consideradas:

- **Spyder:** Entorno de desarrollo *open-source* especializado en la exploración de datos y el análisis científico.
- **Visual Studio:** Herramienta que permite realizar todas las tareas de programación, depuración, pruebas y desarrollo de soluciones para cualquier plataforma.
- **Visual Studio Code:** Versión más ligera y personalizable de Visual Studio.

Herramienta elegida:

- **Visual Studio Code**³

Es el IDE elegido para llevar a cabo el desarrollo de proyecto. Como ventajas principales, presenta un tamaño reducido de instalación respecto a las otras opciones y permite la configuración y ejecución de tareas, además de la capacidad para instalar y personalizar nuevas funcionalidades mediante sus extensiones.

Extensiones utilizadas

Se han escogido una serie de extensiones del *Marketplace* que presenta la herramienta para facilitar la calidad de vida al trabajar con este IDE.

- **Python:** Extensión principal para dar soporte al lenguaje de programación Python para el correcto desarrollo de código (*linting*, formato de código, exploración de variables, depuración, etc.).
- **Python Docstring Generator:** Facilita y Asiste en la creación de comentarios tipo *docstring* para funciones en Python.

³<https://code.visualstudio.com/>

- **Pylance:** Servidor de lenguaje que añade soporte adicional a Python.
- **Trailing Whitespace:** Resalta y recorta los espacios en blanco sobrantes.
- **Visual Studio IntelliCode:** Emplea IA para añadir desarrollo predictivo y autocompletado de código.
- **Docker:** Facilita la creación y gestión de contenedores a través del IDE.

Editor \LaTeX

Herramientas consideradas:

- **Mi \TeX + Texmaker:** Herramientas que realizan la traducción de \LaTeX a texto y permiten gestionar y editar este tipo de archivos, respectivamente.
- **Overleaf:** Plataforma en línea que facilita la gestión y edición de documentos con formato \LaTeX .

Herramienta elegida:

- **Overleaf**

Overleaf es un editor en línea⁴ de \LaTeX . Para utilizarlo no es necesario realizar la instalación de ningún componente, tiene documentación integrada para \LaTeX y permite la visualización de los cambios realizados en tiempo real, además de contar ya con los paquetes más utilizados.

También resulta más cómodo al tratarse de una plataforma *online*, ya que tan solo hace falta disponer de un navegador y conexión a Internet para poder trabajar con ella desde cualquier equipo. Otra de las mejores funcionalidades que ofrece es la posibilidad de comprobar el histórico de los archivos modificados y realizar un *rollback* de los mismos.

Se ha utilizado esta herramienta para elaborar la memoria y los anexos en \LaTeX .

⁴<https://es.overleaf.com/>

Joplin

A lo largo de la duración del proyecto hará falta tomar notas de varios temas diversos. Para facilitar esta tarea, se ha utilizado *Joplin*⁵. Es una plataforma de código abierto que permite gestionar apuntes y notas en forma de *notebooks*.

Entre las principales características que ofrece se encuentra la total privacidad de los datos, la sencilla interfaz que presenta, la facilidad de uso gracias al lenguaje *Markdown* y la sincronización de contenido entre diversos equipos.

Se utilizará principalmente para dejar constancia de los temas comentados durante las reuniones y apuntar información relevante para el proyecto que se vaya encontrando a medida que se desarrolle este trabajo.

Super Productivity

La gestión del tiempo dedicado se ha llevado a cabo mediante la herramienta de código abierto *Super Productivity*⁶. Sus principales funciones consisten en realizar la planificación, seguimiento y gestión de tareas. Permite distribuir tareas a lo largo de diversos proyectos, la asignación de etiquetas personalizadas y tener constancia del tiempo estimado y dedicado para cada una.

Presenta una interfaz sencilla de utilizar y amigable para el usuario que agiliza el trabajo gracias a la utilización de atajos de teclado. Otra de las características más importantes que tiene esta herramienta es la integración con varias plataformas para la importación de tareas. Por lo que la planificación realizada en GitHub y ZenHub se puede extraer a esta herramienta y realizar un mejor seguimiento del tiempo empleado en cada una de ellas.

⁵<https://joplinapp.org/>

⁶<https://super-productivity.com/>

Aspectos relevantes del desarrollo del proyecto

Este apartado pretende recoger los aspectos más interesantes del desarrollo del proyecto, además de la experiencia práctica adquirida durante la realización del mismo con las diversas tecnologías empleadas.

5.1. Extracción de datos

Fuentes de datos

La información necesaria para conseguir el objetivo de este proyecto está formada por opiniones públicas de personas sobre algún tema o temas en concreto. La manera más sencilla de obtener estos datos es empleando recursos web como foros, blogs y redes sociales. Más concretamente, se ha optado por investigar la disponibilidad de *APIs* públicas de los principales sitios web donde las personas publican sus opiniones.

A continuación, se realiza un pequeño resumen de la información de la que se dispone actualmente sobre las *APIs* de cada plataforma:

Twitter

En 2006 se abrió al público la *API REST* [13] de Twitter, que actualmente consta ya de su versión **v2**, aunque coexiste a su vez con algunas partes de la misma aún en la versión **v1.1** y otras de pago (*Premium v1.1* o *Enterprise*). Está basada en *GraphQL*⁷ y devuelve los resultados en formato *JSON*.

⁷Lenguaje de consultas para *APIs* que facilita la gestión de datos y peticiones (<https://graphql.org/>).

Los permisos que se deben asignar son solo de lectura o escritura de contenido. Mientras que el número de peticiones varía en función del *endpoint*, la ventana temporal de restricción se limita a tan solo 15 minutos [15].

Ofrece acceso de lectura, escritura, modificación y borrado de una amplia variedad de recursos, como puede verse en la [Tabla 5.1](#).

Recurso	Versión	Descripción
<i>Tweets</i>	v2 v1.1 <i>Premium</i> <i>Enterprise</i>	Operaciones <i>CRUD</i> .
<i>Users</i>	v2 v1.1 <i>Premium</i> <i>Enterprise</i>	Gestión y búsqueda de usuarios y relaciones entre los mismos.
<i>Spaces</i>	v2	Búsqueda de espacios y participantes.
<i>Direct Messages</i>	v1.1	Envío y respuesta a mensajes directos.
<i>Lists</i>	v2 v1.1	Gestión de listas de contactos.
<i>Trends</i>	v1.1	Identificar tendencias por zonas geográficas.
<i>Media</i>	v1.1	Cargar archivos multimedia.
<i>Places</i>	v1.1	Búsqueda de lugares.

Tabla 5.1: Recursos disponibles a través de la *API* de Twitter

Fuente: [16]

Facebook

La *API* de Facebook originalmente utilizaba *FQL* (*Facebook Query Language*) como lenguaje de consulta, parecido a *SQL*. Sin embargo, en 2010 comenzó la migración hacia *Graph API* [7], actualmente en su versión **v16.0**. Se organiza en función de colecciones, nodos y campos. Un nodo es un objeto único que representa una clase del diccionario de datos en concreto, mientras que los campos son atributos del mismo y una colección comprende un conjunto de nodos. Toda esta información es presentada en formato *JSON*.

Presenta una gran cantidad de permisos [10] que son requeridos para realizar las acciones de gestión, algunos de los cuales es necesario que sean

aprobados por Facebook para su uso. Además, el número de peticiones que se pueden realizar se limita a 200 por hora por cada usuario [11].

La lista de nodos que presenta la *API* es muy extensa, aunque en la [Tabla 5.2](#) se muestran algunos de ellos que podrían resultar útiles para el desarrollo de este proyecto.

Nodo	Descripción
<i>Comment</i>	Comentarios de los objetos.
<i>Link</i>	Enlaces compartidos.
<i>Group</i>	Objeto único de tipo grupo.
<i>Likes</i>	Lista de personas que han dado <i>like</i> a un objeto.
<i>Page</i>	Información sobre páginas.
<i>User</i>	Representación de un usuario.

Tabla 5.2: Muestra de nodos disponibles en *Graph API* de Facebook

Fuente: [8]

Instagram

Lanzada originalmente en 2014 y actualmente integrada junto a la *Graph API* de Facebook. Dispone de dos versiones, una más básica enfocada solamente al consumo de contenido, y la normal, que permite realizar diversos tipos de acciones sobre la cuenta y llevar a cabo su gestión.

Se dispone de un conjunto de permisos requeridos bastante más reducido que para la *API* de Facebook, aunque el límite de peticiones es el mismo ya que funciona sobre la propia *Graph API*.

Al estar basada también en la misma tecnología, la estructura consta de los mismos elementos mencionados en el apartado anterior. La diferencia serían los nodos principales en los que se distribuye su contenido, como se observa en la [Tabla 5.3](#).

Nodo	Descripción
<i>Comment</i>	Comentarios de los objetos.
<i>Hashtag</i>	Representa un <i>hashtag</i> .
<i>Multimedia</i>	Referencia una foto, vídeo, historia o álbum.
<i>User</i>	La cuenta de un usuario.
<i>Page</i>	Información sobre páginas.

Tabla 5.3: Muestra de nodos disponibles en *Graph API* de Instagram

Fuente: [9]

YouTube

Introducida en el año 2013, actualmente en su versión **v3**, y permite la integración de funcionalidades de la plataforma, búsqueda de contenido y análisis demográficos. Cada recurso se representa como un objeto *JSON* sobre el que se pueden ejecutar varias acciones.

Respecto a permisos requeridos, no son tan estrictos como por parte de Meta. No obstante, el número de peticiones se calcula en función de las “unidades” que consume cada tipo de petición, teniendo un total básico de 10.000 al día [3].

Cada recurso se representa como un objeto de datos con identificador único. Entre ellos, los más representativos para la realización de este proyecto podrían ser los expuestos en la **Tabla 5.4**.

Recurso	Descripción
<i>Caption</i>	Representa los subtítulos de un vídeo.
<i>Comment</i>	Comentarios de los objetos.
<i>Playlist</i>	Colección de vídeos accesibles de forma secuencial.
<i>Search result</i>	Información de una búsqueda que apunta a un objeto.
<i>Video</i>	Objeto representativo para un vídeo.

Tabla 5.4: Recursos disponibles a través de la *API* de YouTube

Fuente: [4]

Reddit

Esta *API* fue lanzada en 2011, proporcionando acceso y gestión sobre todas las acciones disponibles desde su interfaz web. También la menos restrictiva de las estudiadas en esta sección, aunque no por ello menos trabajada.

Como ventaja respecto al resto, permite realizar hasta 60 peticiones por minuto [17] a través de todos sus *endpoints*.

Los recursos se representan como objetos tipo *JSON*. En la [Tabla 5.5](#) se pueden observar las principales estructuras de datos.

Recurso	Descripción
<i>Comment</i>	Comentario de las demás estructuras de datos.
<i>Subreddit</i>	Representación de un subforo.
<i>Message</i>	Información sobre mensajes.
<i>Account</i>	Datos de la cuenta de un usuario.

Tabla 5.5: Recursos disponibles a través de la *API* de Reddit

Fuente: [6]

Herramientas

Postman

Es una plataforma para construir y utilizar *APIs* que simplifica el desarrollo y la colaboración. Cuenta con una versión web y una aplicación de escritorio, además de un repositorio público de colecciones de *APIs* y documentación sobre los posibles tipos de llamadas que se les puede realizar.

Esta herramienta será la utilizada para llevar a cabo una inspección inicial de los distintos *endpoints* que presentan las *APIs* investigadas en la sección anterior. Un ejemplo concreto de ello podría ser el siguiente *workspace* de Twitter [12], en el que se presentan documentadas las posibles llamadas a realizar.

API Wrappers

Las interfaces estudiadas en la sección anterior tienen un gran número de usuarios, lo que ha conllevado a la creación de distintas librerías o paquetes en algunos lenguajes de programación que “envuelven” y “atacan” los *endpoints*

de dichas *APIs*. Estos paquetes tienen el objetivo de facilitar al usuario la tarea de crear las consultas y consumir los recursos provenientes de dichas peticiones.

Estas librerías serán utilizadas para construir un primer prototipo para esta etapa de extracción de datos. Más concretamente, se utilizarán las siguientes:

- ***Python Twitter Search API***. Cliente en lenguaje *Python* enfocado en utilizar los *endpoints* de búsqueda de *tweets* [14].
- ***Python-Facebook***. Una librería simple de *Python* que simplifica el uso de la *Graph API* de Meta, dando soporte tanto para Facebook como para Instagram [5].
- ***PRAW: The Python Reddit API Wrapper***. Paquete de *Python* que facilita el acceso a la *API* de Reddit [2].

Airbyte

Esta plataforma permite realizar la creación de un *pipeline* de extracción y guardado de datos de forma sencilla y rápida. Presenta una versión de código abierto y distribuida en contenedores *docker*, junto a una interfaz web que simplifica el proceso de gestión.

Permite la creación, definición y configuración tanto de fuentes de datos como de destinos de los mismos. Actualmente cuenta con más de 300 conectores disponibles para distintas aplicaciones e interfaces web [1].

Esta herramienta es muy probable que sea la que se termine utilizando tras la fase de prototipado para la fase de ingestión de datos de la herramienta final desarrollada.

Trabajos relacionados

En este apartado se describirán otras herramientas similares ya existentes que cumplen un propósito similar al planteado en este proyecto. También se escribirá sobre los principales artículos científicos que comprenden el *state-of-the-art* relacionado con las técnicas de procesamiento de lenguaje natural utilizadas.

6.1. Herramientas similares

Herramientas de pago

Brand24

Es una plataforma⁸ que monitoriza las menciones sobre la marca del cliente tanto en la web como en redes sociales. Utiliza técnicas NLP para analizar en tiempo real los datos de diversas fuentes como blogs, foros, redes sociales, vídeos...

Una de las ventajas competitivas que ofrece es mostrar la influencia que ha tenido cada mención. Como desventaja, cabe destacar el limitado número de menciones que permite monitorizar en sus servicios de suscripción. El rango de precios comprende desde los \$49 mensuales del paquete básico hasta los \$348 del paquete ejecutivo.

⁸<https://brand24.com/>

MonkeyLearn

Es un conjunto de herramientas de análisis de texto que permite crear modelos propios de *machine learning* sobre los datos introducidos, empleando la propia interfaz gráfica de la plataforma.

Como ventaja principal, provee unos modelos ya entrenados que se pueden utilizar en la mayoría de las situaciones, pero permite también entrenarlos sobre los datos específicos que interesen al cliente. Como desventajas, se podrían incluir la manera de establecer la conexión con los datos, puesto que necesita acceso directo a la base de datos del cliente, además de requerir una suscripción mensual de \$299.

Repustate

Es una herramienta de análisis⁹ de sentimientos que analiza de manera sintáctica los datos introducidos para poder evaluar de mejor manera la intención de cada texto. También es capaz de analizar *emojis* según el contexto en el que se utilicen y provee una API que da soporte a 23 idiomas distintos.

Las principales ventajas que ofrece son la gran cantidad de idiomas que soporta y la posibilidad de especificar distintos significados de palabras concretas para mejorar el análisis que realiza. Como principal desventaja, la utilización de este servicio requiere una suscripción mensual de \$199 para su plan *Standard* o \$499 para el *Premium*.

Herramientas gratuitas

Social Searcher

Es una herramienta sencilla¹⁰ que ofrece búsqueda por palabras clave, etiquetas o usuarios y muestra unos análisis básicos sobre los resultados obtenidos. Muestra un *dashboard* con varias pestañas en las que se realizan distintos tipos de análisis, además de gráficos diversos que categorizan las menciones en temas y clasifican las opiniones de los usuarios.

La principal ventaja de esta herramienta es que permite aprovechar sus servicios de manera gratuita y sin límite de consultas, aunque tenga también planes de pago. Como desventaja, las funcionalidades que ofrece la versión gratis son bastante básicas.

⁹<https://www.repustate.com/>

¹⁰<https://www.social-searcher.com/>

Tweet Sentiment Viz

Esta herramienta es la más básica¹¹ de la lista. Muestra una serie de gráficos exploratorios (temas, mapas de calor, nubes de palabras, etc.) sobre los datos buscados en tiempo real en función de palabras clave.

Como principal ventaja, es que funciona bastante bien dentro de unos límites preestablecidos. Entre sus desventajas, esta herramienta analiza únicamente datos de la plataforma Twitter, además de emplear técnicas de bolsas de palabras. Por lo que tendrá dificultades a la hora de interpretar cualquier palabra utilizada que no esté dentro de dichos diccionarios.

6.2. Artículos científicos

BERT: Bidirectional Encoder Representations from Transformers

Se trata de un modelo¹² que utiliza una red neuronal ya entrenada para generar *word embeddings* que son utilizadas posteriormente como características en modelos *NLP*.

BERT se basa en *transformers* (mecanismos de atención que "aprenden" correlaciones entre las palabras de un texto). Estos *transformers* presentan dos componentes, un *encoder* que procesa los datos de entrada y un *decoder* que se encarga de realizar las predicciones correspondientes. Sin embargo, como el objetivo es construir un modelo de lenguaje, tan solo hace falta la primera parte de estos, el codificador.

Mientras que los modelos hasta el momento tomaban una dirección de lectura secuencial de los datos (bien de izquierda a derecha o bien al revés), el codificador del *transformer* es capaz de leer cada palabra del texto a la vez. Esto permite al modelo analizar el contexto general en el que se presenta cada palabra y no teniendo en cuenta solamente una dirección. De esta manera, se considera un modelo "bidireccional", aunque en realidad no tenga una dirección como tal.

Generalmente, los modelos de lenguaje se entrenan intentando predecir una secuencia de palabras dentro de un texto, lo que los convierte en unidireccionales. Por ello, *BERT* emplea dos estrategias para mantener su habilidad bidireccional:

¹¹https://www.csc2.ncsu.edu/faculty/healey/tweet_viz/tweet_app/

¹²<https://arxiv.org/abs/1810.04805>

- ***Masked LM (MLM)***. La primera estrategia que se utiliza es ocultar mediante un *token* a forma de máscara aproximadamente un 15 % de las palabras del texto de entrada del codificador. Posteriormente, el modelo intentará predecir las palabras que faltan basándose en el contexto que las rodea.
- ***Next Sentence Prediction (NSP)***. La segunda estrategia consiste en entrenar el modelo mediante pares de frases. La mitad de los datos de entrada se divide de tal manera que la segunda frase de cada par es la que va a continuación de la primera frase en el texto original. Mientras que en la otra mitad de los datos la segunda frase se escoge al azar del texto original. De tal manera, se asume que el modelo será capaz de distinguir correctamente qué frase tiene sentido a continuación de otra. Se utilizan una serie de *tokens* para indicar el inicio y final de cada frase.

Ambas estrategias se ponen en práctica y se entrenan a la vez para conseguir minimizar la *loss function* del modelo.

Conclusiones y Líneas de trabajo futuras

Todo proyecto debe incluir las conclusiones que se derivan de su desarrollo. Éstas pueden ser de diferente índole, dependiendo de la tipología del proyecto, pero normalmente van a estar presentes un conjunto de conclusiones relacionadas con los resultados del proyecto y un conjunto de conclusiones técnicas. Además, resulta muy útil realizar un informe crítico indicando cómo se puede mejorar el proyecto, o cómo se puede continuar trabajando en la línea del proyecto realizado.

Apéndice

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En las siguientes secciones se realizará un estudio de la planificación temporal seguida durante el desarrollo de este proyecto, además de la viabilidad tanto económica como legal que podría llegar a suponer este trabajo.

Debido a la naturaleza inherente del proyecto, al no tratarse de un *software* típicamente tradicional sino más bien centrado hacia la investigación e implementación de modelos de *machine learning*, no ha resultado sencillo llevar a cabo algunas de las buenas prácticas y conceptos normales de acuerdo a un “Plan de Proyecto Software” tradicional.

A.2. Planificación temporal

La planificación del proyecto se ha llevado a cabo mediante la metodología de desarrollo ágil *Scrum*. A continuación se realiza un desglose de los distintos *Sprints* llevados a cabo.

Inicialmente, se presentan las tareas correspondientes a cada iteración del trabajo y su duración inicial estimada. Posteriormente, se realiza una comparación entre el tiempo total estimado y el real gastado mediante la ilustración de gráficos *burn-down*.

Sprint 0 (01/02/2023 - 15/02/2023)

Este *Sprint* inicial será dedicado a la preparación del entorno de trabajo para el proyecto. Se elegirán las herramientas con las que se trabajará en algunas de las etapas del proyecto, se investigarán técnicas y librerías a utilizar, se realizarán unas pruebas concepto iniciales y se comenzará la labor de documentación.

- **Gestión del *Sprint* (4h).** Se realizará el planteamiento de las tareas a llevar a cabo a lo largo de este sprint y se documentarán en el apartado **A.2 - Planificación temporal del Apéndice A - Plan de Proyecto Software** de los anexos del proyecto.
- **Elegir IDE (2h).** Para la realización de este proyecto será necesaria la utilización de diversos lenguajes de programación, por lo que la elección de un entorno de desarrollo integrado adecuado resultará de gran ayuda.
- **Estudiar guía \LaTeX (2h).** Como objetivo para la generación de la memoria del proyecto, se va estudiar una guía sobre \LaTeX con el fin de recordar los conocimientos necesarios para poder crear la documentación correspondiente.
- **Documentación de la memoria - Técnicas y herramientas (4h).** Comenzar con la documentación de la memoria del proyecto, con la sección “Técnicas y herramientas”. De manera inicial, se documentará lo siguiente:
 - **Técnicas**
 - *Scrum*
 - *Natural Language Processing*
 - *Sentiment Analysis*
 - **Herramientas**
 - GitHub
 - ZenHub
 - Overleaf
 - Joplin
 - Super Productivity

- **Documentación de la memoria - Trabajos relacionados (4h).**

La siguiente parte de la memoria que se va a redactar será la sección de “Trabajos relacionados”. En este apartado se describirán otras herramientas similares ya existentes que cumplen un propósito similar al planteado en este proyecto.

También se escribirá sobre los principales artículos científicos que comprenden el *state-of-the-art* relacionado con las técnicas de procesamiento de lenguaje natural que serán utilizadas.

- **Investigar y probar recursos NLP ya existentes (8h).** Ya que inicialmente no se prevé el desarrollo de un algoritmo NLP propio, se investigará el *state-of-the-art* sobre análisis de sentimientos y se comprobará si existen recursos ya implementados para utilizar en el proyecto.

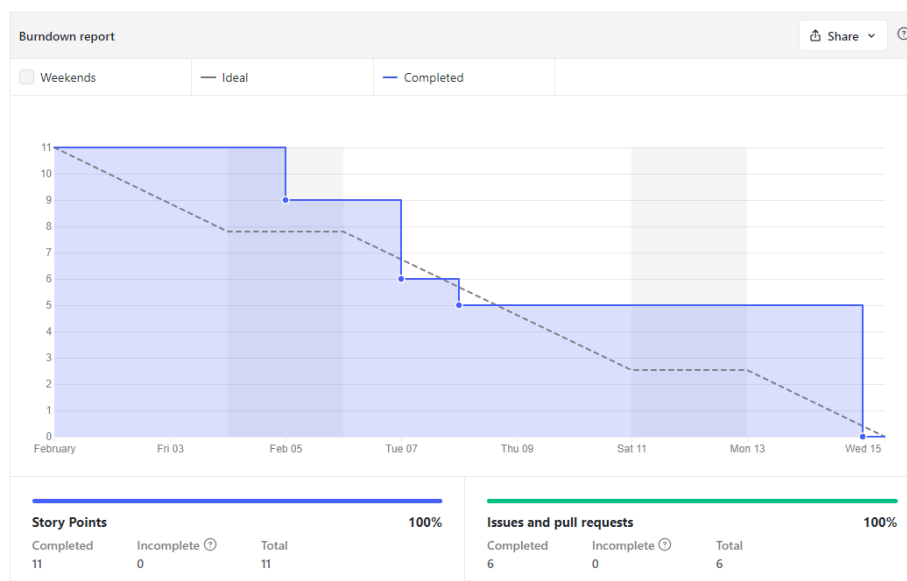


Figura A.1: Gráfico *burn-down* del *Sprint 0*

***Sprint 1* (15/02/2023 - 01/03/2023)**

Durante la duración de este *Sprint* se investigarán las *APIs* de las posibles plataformas de las que se va a extraer la información textual y las herramientas disponibles para realizar la primera etapa del proceso *ETL*. También se comenzará a realizar una primera prueba concepto utilizando los recursos investigados.

- **Gestión del *Sprint* (2h).** Se realizará el planteamiento de las tareas a llevar a cabo a lo largo de este sprint y se comenzará a documentar la sección **Aspectos relevantes**.
- **Elegir IDE (2h).** Para la realización de este proyecto será necesaria la utilización de diversos lenguajes de programación, por lo que la elección de un entorno de desarrollo integrado adecuado resultará de gran ayuda.
- **Investigar posibles APIs a utilizar (4h).** Como se ha especificado en el objetivo de este proyecto, se necesita información y opiniones públicas de las que poder obtener conocimiento sobre temas concretos. Para ello, se investigará la existencia de *APIs* públicas de los principales sitios web en los que la gente suele expresar sus opiniones de manera general, siendo estos los foros, blogs y redes sociales.
- **Escoger tema inicial con alta polaridad (2h).** Para comprobar el correcto funcionamiento de los recursos *NLP* investigados en el *Sprint* anterior, se escogerá un tema con alta polaridad sobre el que se centrarán las pruebas de dichos recursos. De esta manera, será más sencillo de visualizar el correcto funcionamiento de estos y el análisis de sentimientos mediante ejemplos claros.
- **Investigar herramientas para realizar la extracción de datos (8h).** En este punto se comenzará a investigar las posibles herramientas para realizar la etapa de extracción de datos del proyecto. Para ello, se compararán las principales alternativas disponibles para realizar la recogida de información de los recursos web descubiertos en este mismo *Sprint*.
- **Crear prototipo inicial para la etapa de extracción de datos (8h).** Para realizar una mejor comparación de las herramientas investigadas en la tarea anterior, se creará un pequeño prototipo para esta primera etapa de extracción de datos sobre las *APIs* seleccionadas, empleando para ello las tecnologías escogidas más relevantes.
- **Documentar los procesos del sprint actual (8h).** A lo largo de este sprint se va a realizar la investigación de varios recursos que deberán ser correctamente documentados, ya que las siguientes etapas del proyecto dependerán de la calidad de la información proporcionada inicialmente.

A.3. Estudio de viabilidad

Viabilidad económica

Viabilidad legal

Apéndice B

Especificación de Requisitos

- B.1. Introducción
- B.2. Objetivos generales
- B.3. Catalogo de requisitos
- B.4. Especificación de requisitos

Apéndice C

Especificación de diseño

- C.1. Introducción
- C.2. Diseño de datos
- C.3. Diseño procedimental
- C.4. Diseño arquitectónico

Apéndice D

Documentación técnica de programación

- D.1. Introducción
- D.2. Estructura de directorios
- D.3. Manual del programador
- D.4. Compilación, instalación y ejecución del proyecto
- D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

- E.1. Introducción
- E.2. Requisitos de usuarios
- E.3. Instalación
- E.4. Manual del usuario

Bibliografía

- [1] Airbyte. Hundreds of connectors out-of-the-box. [En línea]. Airbyte, Inc., 2023. Disponible en <https://airbyte.com/connectors>, 2023. [Fecha de consulta: 28-02-2023].
- [2] Bryce Boe. Praw: The python reddit api wrapper. [En línea]. reddit, inc., 2023. Disponible en <https://github.com/praw-dev/praw>, 2023. [Fecha de consulta: 28-02-2023].
- [3] Google Developers. Quota usage. [En línea]. YouTube Data API, 2023. Disponible en <https://developers.google.com/youtube/v3/getting-started#calculating-quota-usage>, 2023. [Fecha de consulta: 26-02-2023].
- [4] Google Developers. Resource types. [En línea]. YouTube Data API, 2023. Disponible en <https://developers.google.com/youtube/v3/docs#resource-types>, 2023. [Fecha de consulta: 26-02-2023].
- [5] IkarosKun. Python-facebook. [En línea]. GitHub, 2023, 2023. Disponible en <https://github.com/sns-sdks/python-facebook>, 2022. [Fecha de consulta: 28-02-2023].
- [6] Reddit Inc. Api overview. [En línea]. reddit, inc., 2023. Disponible en <https://www.reddit.com/dev/api>, 2023. [Fecha de consulta: 26-02-2023].
- [7] Meta. General information - graph api. [En línea]. Meta for Developers, 2023. Disponible en <https://developers.facebook.com/docs/graph-api/overview>, 2023. [Fecha de consulta: 26-02-2023].

- [8] Meta. Graph api reference. [En línea]. Meta for Developers, 2023. Disponible en <https://developers.facebook.com/docs/graph-api/reference>, 2023. [Fecha de consulta: 26-02-2023].
- [9] Meta. Instagram graph api reference. [En línea]. Meta for Developers, 2023. Disponible en <https://developers.facebook.com/docs/instagram-api/reference>, 2023. [Fecha de consulta: 26-02-2023].
- [10] Meta. Permissions reference. [En línea]. Meta for Developers, 2023. Disponible en <https://developers.facebook.com/docs/permissions/reference>, 2023. [Fecha de consulta: 26-02-2023].
- [11] Meta. Rate limits - graph api. [En línea]. Meta for Developers, 2023. Disponible en <https://developers.facebook.com/docs/graph-api/overview/rate-limiting>, 2023. [Fecha de consulta: 26-02-2023].
- [12] Twitter. Twitter's public workspace. [En línea]. Postman, Inc., 2023. Disponible en <https://www.postman.com/twitter/workspace/twitter-s-public-workspace/collection/9956214-784efcda-ed4c-4491-a4c0-a26470a67400?ctx=documentation>, 2023. [Fecha de consulta: 28-02-2023].
- [13] Inc. Twitter. Getting started with the twitter api. [En línea]. Twitter, 2023. Disponible en <https://developer.twitter.com/en/docs/twitter-api/getting-started/about-twitter-api>, 2023. [Fecha de consulta: 26-02-2023].
- [14] Inc. Twitter. Python client for the twitter api v2 search endpoints. [En línea]. GitHub, 2023. Disponible en <https://github.com/twitterdev/search-tweets-python/tree/v2>, 2023. [Fecha de consulta: 28-02-2023].
- [15] Inc. Twitter. Rate limits. [En línea]. Twitter, 2023. Disponible en <https://developer.twitter.com/en/docs/twitter-api/rate-limits#v2-limits>, 2023. [Fecha de consulta: 26-02-2023].
- [16] Inc. Twitter. Twitter api platform resources. [En línea]. Twitter, 2023. Disponible en <https://developer.twitter.com/en/docs/twitter-api/getting-started/about-twitter-api#item2>, 2023. [Fecha de consulta: 26-02-2023].
- [17] Josh Wardle. Wiki api. [En línea]. GitHub, 2015. Disponible en <https://github.com/reddit-archive/reddit/wiki/API>, 2023. [Fecha de consulta: 26-02-2023].