ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA

Trabajo Fin de Grado

DISEÑO DE UN CONTROL DE CONTENIDOS "PROANAMIA" MEDIANTE UNA EXTENSIÓN DE NAVEGADOR

(Design of a "ProAnaMia" content control through a browser extension)

Para acceder al Título de

Graduado en

Ingeniería de Tecnologías de Telecomunicación

Autor: Jorge Cases López
Septiembre -2025

UC | ETS de Ingenieros Industriales y de Telecomunicación

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

CALIFICACIÓN DEL TRABAJO FIN DE GRADO

Realizado por: Jorge Cases López
Director del TFG: Alberto Eloy García Gutierrez
Título: "DISEÑO DE UN CONTROL DE CONTENIDOS "PROANAMIA" MEDIANTE UNA EXTENSIÓN DE NAVEGADOR "
Title: "Design of a "ProAnaMia" content control through a browser extension "
Presentado a examen el día: 17 de septiembre de 2025

para acceder al Título de

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Composición del Tribunal:
Presidente (Apellidos, Nombre): PEREZ ARRIAGA, JESUS
Secretario (Apellidos, Nombre): DIEZ FERNANDEZ, LUIS FRANCISCO
Vocal (Apellidos, Nombre): PRIETO TORRALBO, PABLO

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo: El Presidente **Fdo: El Secretario**

Fdo: El Vocal **Fdo: El Director del TFG**
(sólo si es distinto del Secretario)

Vº Bº del Subdirector Trabajo Fin de Grado Nº
(a asignar por Secretaría)

Resumen

En este trabajo se ha realizado una extensión de Google Chrome utilizando un modelo de machine learning para la detección y posterior bloqueo en tiempo real de los contenidos “ProAnaMia” (apología de la anorexia y la bulimia). Para ello, se ha complementado el servidor que integra los modelos de clasificación y procesa las peticiones de análisis del texto que se extrae de las páginas web visitadas. La extensión, desarrollada desde cero, intercepta de manera dinámica el contenido de cada sitio que se visita, envía el texto al servidor para que lo evalúe y, en caso de que el contenido sea peligroso, restringe al usuario la web redirigiéndolo a una página de bloqueo. Además, tiene una lista negra con todas las direcciones de los sitios peligrosos almacenada de manera local, para su posterior revisión en caso de que fuese necesario. Este sistema ofrece así una solución completa a un problema en auge dentro de internet.

Abstract

This project presents the development of a Google Chrome extension that leverages a machine learning model to detect and block “ProAnaMia” content (glorification of anorexia and bulimia) in real time. A complementary server was implemented to host the classification models and handle analysis requests for text extracted from visited web pages. The extension, built from scratch, dynamically intercepts page content, sends it to the server for evaluation, and—if deemed dangerous—redirects the user to a designated block page. Additionally, it maintains a locally stored blacklist of all blocked URLs for subsequent review if needed. This modular system provides a comprehensive solution to counter the growing problem of harmful online content.

Agradecimientos

En primer lugar, deseo agradecer a Isa, que me ha acompañado durante todo el grado animándome y ayudándome siempre que podía. Sin su respaldo, este trabajo y el grado no habrían sido posibles.

También quiero agradecer a mi familia y amigos por estar ahí siempre que los he necesitado. Gracias a su apoyo he logrado ir sacando este grado y trabajo.

Por último, quiero agradecer a mi tutor del TFG Alberto Eloy García por haberme ayudado desde que llegue a la Universidad de Cantabria y haberme propuesto este tema. Su ayuda y paciencia han sido fundamentales para la realización de este Trabajo de Fin de Grado.

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

ACRÓNIMOS

TCA: Trastornos de la Conducta Alimentaria

UE: Unión Europea

HTML: HyperText Markup Language

CSS: Cascading Style Sheets

JSON: JavaScript Object Notation

DOM: Document Object Model

API: Application Programming Interface

IDE: Integrated Development Environment

TFG: Trabajo de Fin de Grado

URL: Uniform Resource Locator

IA: Inteligencia Artificial

BERT: Bidirectional Encoder Representations from Transformers

KNN: K-Nearest Neighbours

MNBC: Multinomial Naive Bayes

LRC: Logistic Regression

NGFW: Next-Generation Firewall

DPI: Deep Packet Inspection

IDS: Intrusion Detection Services

IPS: Intrusion Prevention Systems

IDP: Intrusion Detection and Prevention

UTM: Unified Threat Management

1 INTRODUCCIÓN

El uso masivo de internet y de las redes sociales ha transformado de manera radical el modo en el cual las personas acceden y comparten cualquier tipo de información. Aunque el fin de estas herramientas sea el facilitar el acceso a la información y que la comunicación sea más sencilla, también ha provocado el que contenidos dañinos que pueden provocar comportamientos perjudiciales para la salud sean más accesibles. En específico, la subcultura de internet "ProAnaMia", que promueve la anorexia y la bulimia como estilos de vida saludables y recomendados, se ha extendido de manera alarmante entre personas vulnerables, especialmente en adolescentes, aprovechando la facilidad que estas herramientas proveen para compartir información.

Es cierto que los proveedores de servicios intentan aplicar filtros y moderar el contenido de la web, pero los métodos tradicionales actualmente no tienen la capacidad de frenar esta nueva tendencia por culpa de la evolución de este lenguaje críptico y de la carga dinámica de la información.

Este TFG propone una solución proactiva y en tiempo real para detectar y bloquear de manera automática todas las páginas web que contengan este tipo de contenidos mediante una extensión de Google Chrome que utiliza un servidor de clasificación de texto que utiliza modelos de machine learning. Está diseñada específicamente para mejorar la accesibilidad y el uso de las nuevas tecnologías. La aplicación ofrecerá una interfaz intuitiva y herramientas de asistencia que permitirán a los usuarios navegar y utilizar tecnologías modernas de manera más eficiente y efectiva.

1.1 Contexto y Motivación

En la actualidad el acceso a contenidos en internet está disponible para personas de todas las edades. Esto significa que están expuestas de manera repetida a un flujo continuo de imágenes e ideas relacionadas con los estereotipos actuales de belleza. El consumo de las redes sociales ha crecido de manera exponencial en los últimos años y en España, actualmente el número de usuarios activos en redes sociales es de más de 32,4 millones.[1] Además, el uso de internet es mayor en los jóvenes de 16 a 24 años, ya que utilizan este el 99,6% de los hombres y el 99,9% de las mujeres. Los resultados de uso de internet en España en 2024 son mayores que en la UE. En España el porcentaje de personas entre los 16 y 74 años que usan internet es del 95,4% en hombres y 96,2% en mujeres, mientras que en la UE son del 93,1% en hombres y 92,6% en mujeres.[2]

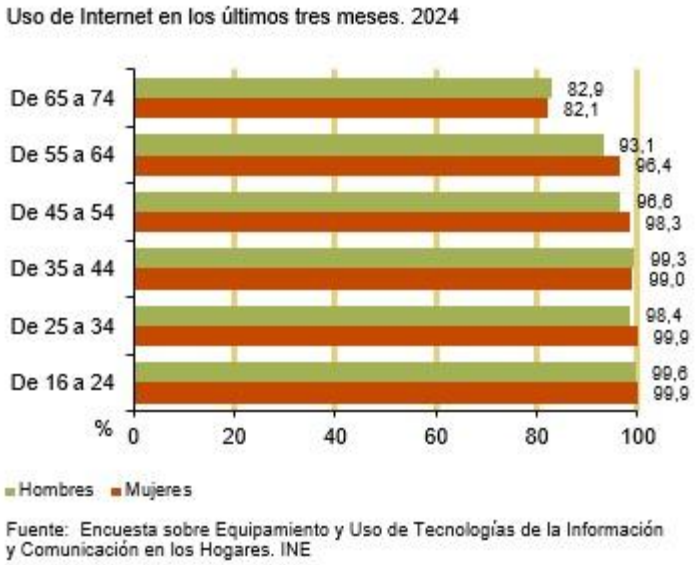


Ilustración 1 Uso de internet últimos 3 meses de 2024 en España

Instagram y TikTok se encuentran dentro de las tres redes sociales más utilizadas en la actualidad y cuentan unas cifras de usuarios activos mensuales de 2.000 millones y 1.590 millones respectivamente [3]. En España, la generación Z lidera el uso de redes sociales con 1h y 10min diarios [4]. En Estados Unidos, los usuarios activos en Instagram son el 72% de los adolescentes y el 76% de los jóvenes entre dieciocho y veintinueve años.[5]

Debido a esto, la creciente facilidad para acceder y compartir contenido "ProAnaMia" ha provocado una gran preocupación. Ante esta realidad, compañías como Instagram [6] o TikTok [7] han adoptado medidas contra este problema censurando o limitando cualquier contenido que promueva o incentive estas conductas nocivas con el objetivo de proteger la salud de las personas que usen sus plataformas. Además, si alguna publicación relacionada consigue traspasar los filtros puestos por la aplicación, los usuarios pueden reportarlo a las aplicaciones para que se bloquee.

Aun teniendo todos estos filtros, el contenido "ProAnaMia" en la red esta muy extendido y consigue traspasar los filtros puestos por estas compañías con bastante facilidad.

Las mejores soluciones hasta el momento son el poner un control parental al dispositivo, bloqueando de esta manera las aplicaciones que puedan generar problemas y reducir el tiempo de uso de los usuarios, poner filtros de contenido para bloquear contenidos web como el Norton Family o el Kaspersky Safe Kids [8] o usar aplicaciones de monitoreo, que permiten ver lo que se está haciendo en los dispositivos y bloquearlo de manera remota.

Todas estas soluciones no terminan de solucionar el problema ya que o son muy genéricas y no pueden filtrar el contenido de manera eficiente o tienes que estar controlando 24h al día el contenido.

Lo expuesto anteriormente destaca la importancia de implementar medidas preventivas eficaces que restrinjan el acceso a este tipo de contenidos, especialmente a personas que sean propensas a este tipo de trastornos.

A partir de un servidor programado en Python el cual gracias a técnicas de "machine learning" es capaz de comprobar la información ordenándola en diferentes categorías como, por ejemplo: belleza, patología, medioambiente, deportes… se va a desarrollar una extensión de Google Chrome la cual categorice la información y de esta manera bloquee el contenido detectado como "ProAnaMia". Este servidor es un desarrollo del Director de este trabajo, el profesor Alberto Eloy García, el cual ha permitido que sea utilizado en este trabajo el mismo.

Se realizará una extensión del navegador Google Chrome en vez de un navegador personalizado que podría ser idealmente la mejor opción porque que la mayor parte de la población en España usa este navegador y es más complicado que empiecen a usar otro a que se descarguen una extensión, ya que la mayoría de las personas están habituadas a trabajar con ellas por su simplicidad.

1.2 OBJETIVOS

Estudio de la aplicación del modelo de machine learning para la clasificación de contenidos "ProAnaMia" (apología de la anorexia y bulimia) para su uso en tiempo real sobre navegadores web. Como condición fundamental, el formato menos intrusivo para la privacidad de los datos de los usuarios es integrar el resultado del clasificador y la función de control en el mismo navegador mediante una extensión, evitando la captura de todo el tráfico del usuario. De esta forma, el usuario y/o tutor legal solo necesitan cumplimentar la aceptación de su uso.

Para llevar a cabo lo expuesto anteriormente, es necesario realizar la actualización e instalación del servidor, dejándolo disponible para recibir las peticiones de comprobación desde una posible extensión de navegador, para lo cual se definen las siguientes necesidades:

- Es deseable que un menú PopUp de la extensión permitan gestionar tanto el número como las URLs bloqueadas. Estas webs serán almacenadas de manera local en el dispositivo y se mostrarán en la extensión para que, de esta manera, los padres o la persona que esté llevando el control

puedan utilizarlas para añadirlas en una aplicación de control de contenido de otro dispositivo que no tenga esta extensión instalada. Además, es necesario aportar el correspondiente feedback al usuario, ya sea porque se piense que la página a la cual se quiere acceder es un falso positivo, negativo, porque exista un error técnico o solo para enviar una sugerencia al administrador.

- La extensión debe de bloquear todo el contenido detectado, por cualquiera de los tres modelos que revisa el servidor, como “ProAnaMia” con un porcentaje mayor al 5%. En ese momento, se redirigirá a una página web de bloqueo en la cual se explique al usuario el por qué se ha bloqueado ese contenido y de esta manera el usuario no pueda ver el contenido peligroso.

2 MARCO TEÓRICO

El presente capitulo tiene como objetivo establecer las bases teóricas necesarias para el desarrollo de este trabajo. Se analizarán las tecnologías actuales de detección de contenidos, tanto las tradicionales como la evolución de estas gracias a la inteligencia artificial. Además, se examinará la problemática "ProAnaMia" y su impacto en personas vulnerables junto con una revisión de las extensiones de Google como herramientas de control de contenido. Por último, se presentará el servidor de clasificación utilizado en este TFG.

2.1 TECNOLOGÍAS DE DETECCIÓN DE CONTENIDO NOCIVO

En 2025, las técnicas de detección de contenido nocivo han evolucionado mucho, pasando de filtros manuales y basados en palabras clave a sistemas de IA capaces de analizar texto, imagen y vídeo. La capacidad de tener modelos con un aprendizaje más profundo brinda la capacidad de detectar contenidos que anteriormente no se podían abordar.

Los sistemas actuales cuentan con una adaptación continua ya que los modelos se reentrenan de manera continua, son capaces de detectar las infracciones en tiempo real, cosa que es vital ya que en transmisiones en vivo y redes sociales si no se es capaz de detectar en tiempo real no se podrá filtrar de manera correcta. Además de que se puede incluir como ya se expuso antes, texto, imágenes, audios o videos. Todo esto tiene una escalabilidad muy grande y es la razón de este TFG, aprovecharse de los nuevos avances para mejorar cosas que antes no se podían hacer de manera tan eficiente y eficaz.

2.1.1 Sistemas de filtrado tradicionales

Los sistemas tradicionales de control de contenidos más usados históricamente han sido: los firewalls, los servidores proxi y los sistemas de DNS filtering.

A) Firewalls

Los firewalls de última generación, NGFW, han evolucionado de manera considerable. Los tradicionales solamente inspeccionaban de manera simple los puertos y protocolos en la capa de transporte, pero, los NGFW modernos, han desarrollado una inspección profunda de paquetes (DPI) que opera en la capa de aplicación del modelo OSI. [9]

Las nuevas prestaciones de los firewalls son:

- Filtrado de URLs: Permite bloquear el contenido o las páginas web que el administrador quiera para que el usuario final no pueda acceder a ellas.
- Inspección Profunda de Paquetes (DPI): Esta es una herramienta que analiza y monitorea el tráfico de internet. Con esta tecnología se puede detectar, identificar y categorizar la información de internet que el filtrado tradicional de paquetes no puede hacer. Para poder bloquear un tipo de información determinada debemos proveerle de ejemplos para que el filtrado sea más efectivo, ya sea el tipo de paquete, palabras clave, etc.[10]
- Detección y prevención de intrusiones: Los nuevos firewalls pueden detectar ciertos patrones o anomalías que tienden a ser los pasos previos a un ciberataque y de esta manera bloquearlos para proteger al usuario. Estas tecnologías son IDS, IPS y IDP.
- Gestión unificada de amenazas (UTM): Es una tecnología que combina en el firewall varias funciones de seguridad esenciales para proteger las redes.

A pesar de estos avances, las redes siguen teniendo muchas limitaciones en el momento en el que se trabaja con contenido dinámico pierden mucha eficacia. Y esto se acentúa en gran medida en plataformas como las redes sociales donde el contenido se genera de manera continua.

B) Servidores Proxi

Los servidores proxi generan una solución eficiente ya que se encuentran entre el usuario y el servidor de destino y de esta manera, se consigue solucionar el problema de que el contenido de internet este cifrado. Actúa como intermediario entre el usuario y el servidor y cuando el usuario busca cualquier información en internet, en lugar de pedírsela directamente al servidor de destino, la petición pasa primero por el proxi y este hace la petición al destino. Para el tráfico desde el servidor de destino hasta el usuario, la información también pasa por el servidor proxi. De esta manera se consigue que el servidor proxi pueda ver el trafico en claro sin cifrar y puede filtrar el contenido en caso de que sea necesario conociendo en todo momento la información que se envía al usuario.

Este tipo de arquitectura permite:

- Filtrar en tiempo real: El contenido de las webs a las que el usuario quiere acceder, pasa primero por el proxi, por lo que es sencillo bloquear el contenido en caso de que sea necesario antes de que llegue al primero.
- Análisis de contenido: A estos servidores se le pueden implementar herramientas que nos ayuden a analizar el contenido tanto que solicita el usuario como el que proporcionan las páginas web.

Podría parecer que esta es la solución perfecta para el control de contenido, pero existen algunos problemas con este método. El primero es que puede comprometer la seguridad de las comunicaciones ya que el funcionamiento de un proxi es, en esencia, similar al de un "man-in-the-middle" y los navegadores modernos pueden identificarlo como un fallo de seguridad. Además, puede generar latencia por culpa de tener que pasar toda la información por el proxi y ser analizada y si no se configura de manera correcta, se es más susceptible a ciberataques.[11]

C) DNS Filtering

El filtrado de DNS es mucho más rápido que los otros ya que no necesita analizar todo el tráfico de la red, únicamente intercepta consultas DNS y con ello puede:

- Bloquear dominios: Se realizar una blacklist con los dominios a los que se quiere evitar el acceso.
- Redirige: Si se intenta acceder a un dominio que se encuentra en la blacklist, se redirige a una página de aviso.
- Categorizar dominios: Puede usar algoritmos de IA para categorizar los nuevos dominios de manera automática. Antes del desarrollo de la IA esto no era posible.

Pero como en los métodos anteriores, siempre existen fallos o puertas traseras para saltarse estos filtros. Si el usuario cambia el DNS y deja de usar el que se tiene programado, el filtro no sirve de nada. El usuario únicamente tiene que cambiar su servidor DNS a otro, como por ejemplo a los de Google que son públicos y de esta manera evita el filtro.

2.1.2 Evolución hacia la Inteligencia Artificial

Como ya se comentó con anterioridad, la llegada de la inteligencia artificial ha ayudado mucho a desarrollar nuevas habilidades técnicas lo cual ha impulsado una gran evolución tecnológica.

Gracias a esta, se han generado soluciones modernas que se aprovechan de ella y pueden:

- Detectar patrones complejos: el filtrado ya no es únicamente de palabras clave, se puede hacer gracias a que es capaz de detectar patrones mucho más complejos.

- Adaptarse de manera dinámica: el lenguaje va evolucionando de manera continua y la IA es capaz de aprender dinámicamente nuevas formas de expresarse y los nuevos intentos de evasión.

2.2 PROBLEMÁTICA "PROANAMIA"

"ProAnaMia" es el término relativo a una subcultura de internet que promueve la anorexia (Ana) y la bulimia (Mia) como modos de vida saludables en vez de TCAs, que son trastornos mentales graves que requieren de la ayuda de un profesional sanitario para solventarlos. Gracias a la globalización de internet y el auge de las redes sociales esta tendencia ha aumentado de manera preocupante.

Estas prácticas se encuentran en todo tipo de páginas web, no solamente en redes sociales donde ahora mismo están teniendo un crecimiento preocupante. Están incluso en las que parecen más inofensivas, como en páginas en las cuales los usuarios escriben novelas y relatos cortos donde hay relatos para niños y adultos como es watppad.com y otras del estilo.

Los usuarios de estas comunidades comparten contenido dañino como:

- Consejos de restricción alimentaria: estos imponen restricciones extremas y técnicas para poder mantener un ayuno prolongado.
- Métodos de purga: proporcionan información para el uso abusivo e irregular de laxantes y diuréticos.
- "Thininspiration": son imágenes y videos que alaban la delgadez extrema para motivar e inspirar a las personas.
- Competencias de pérdidas de peso: existen competiciones en las cuales los usuarios ponen sus pesos y compiten contra otros perdiendo peso de manera peligrosa.
- Rechazo a la ayuda médica: fomentan que los usuarios no busquen ayuda sanitaria ni busquen otras maneras de recuperación.

La investigación científica ha documentado de manera extensa el impacto nocivo de la exposición de los usuarios a contenido "ProAnaMia". Estudios profesionales han demostrado que la exposición a este tipo de contenidos, aunque sea mínima, produce cambios clínicamente significativos en el comportamiento alimentario de las personas. [12]

Se ha demostrado que los usuarios frecuentan este contenido:

- Reducen las calorías ingeridas hasta en un 80%
- Aprenden nuevas estrategias para controlar su peso
- Aumenta el riesgo de comportamientos autolesivos y de aislamiento social
- Adquieren mayor resistencia a buscar ayuda profesional

Además, estos estudios demuestran que las chicas adolescentes son las más vulnerables para este tipo de contenidos y esto puede provocar daños psicológicos graves durante el desarrollo psicológico de estas. [13]

Como ya se comento en este trabajo, se están desarrollando filtros para estos tipos de contenidos, pero, a la par, las comunidades “ProAnaMia” desarrollan nuevo vocabulario críptico para que estos no funcionen.

Se está creando una terminología encubierta para que los filtros tradicionales no les detecten usando, por ejemplo:

- “Ana” y “Mia” como términos equivalentes a la anorexia y la bulimia respectivamente.
- “Princesas” y “Príncipes” para referirse a las personas que componen estos grupos.
- “Mariposas”, “libélulas” y “ángeles” como categorías de pertenencia dentro de estos grupos.
- “thinspo” en lugar de “thininspiration”
- Cambio de letras por números para evitar filtros como 4na y m1a

Con los métodos actuales existen varias limitaciones a la hora de bloquear este tipo de contenido. Los filtros actuales:

- No son capaces de distinguir entre contenido médico de la categoría y el contenido malicioso promocional de los trastornos.
- Dependen de listas de palabras que quedan rápidamente obsoletas.

También, redes sociales como Instagram, TikTok y Facebook han desarrollado políticas en contra del contenido “ProAnaMia” pero estas políticas son reactivas y dependen de reportes de los usuarios para identificar este contenido. Además, estas comunidades migran de manera completa a otras plataformas con menos moderación.

2.3 Extensiones de Google Chrome para Detección de Contenidos

El navegador más usado en España es Google Chrome con un 61,23% de los usuarios de internet utilizándolo.[14] Y el uso de las extensiones de navegadores está cada vez más extendido ya que gracias a ellas podemos personalizar a nuestro gusto los navegadores.

Las extensiones de Chrome para detección de contenidos nocivos han evolucionado a pasos agigantados gracias a la IA y al procesamiento multimodal el cual nos permite inspeccionar tambien imágenes y videos. Estas se usan también para detectar deepfakes e imágenes alteradas que cada vez son más complicadas de detectar, pero también tenemos modelos mejor entrenados para detenerlos.

Una de las ventajas de Google Chrome es que puedes procesar los datos de manera local y de esta manera mantienes tu privacidad.

El reto más significativo al cual se enfrentan las extensiones es al de la latencia y la escalabilidad. Cada vez más contenido en la web es en vivo y si quieres desarrollar una extensión no puedes quedarte atrás en este aspecto ya que la velocidad a la hora de conseguir las cosas en la web cada vez coje más importancia.

La estructura de las extensiones de Google Chrome es la siguiente:

2.3.1 Manifest.json

Este archivo actúa como el núcleo de la extensión, definiendo metadatos esenciales, permisos requeridos y recursos utilizados. Gracias a la versión 3 se ha mejorado mucho tanto la seguridad como el rendimiento. Algunas de estas mejoras son: [15]

- Gestión granular de permisos: esta característica nos permite acceder a recursos específicos.
- Service workers: ahora es obligatorio usar un service worker que permita realizar labores en segundo plano mejorando la eficiencia y reduciendo el consumo de recursos.
- APIs declarativas [16]: cambia de webRequest a la nueva API declarativeNetRequest que es más restrictiva para bloquear contenidos.
- Política de seguridad de contenido: esta es más restrictiva para prevenir vulnerabilidades.

2.3.2 Background Scripts

Los background sripts son el cerebro de la extensión. Manejan los eventos de Chrome y coordinan la interacción entre los archivos. Con la versión 3, estos archivos también han evolucionado y los nuevos cambios son:

- Se activan bajo demanda: los service workers se activan cuando se solicita su uso y esto genera un menor consumo del sistema.
- Mantienen el estado: en todas las pestañas y ventanas en las que se trabaje se mantiene el mismo estado.
- Gestionan la comunicación: se encargan de que las APIs de Chrome estén en contacto con los servidores externos.
- Procesan eventos: detectan los nuevos eventos que surgen al recargar una página, cambiar la URL, etc.

2.3.3 Content Scripts

Los content scripts se encargan de interactuar con el DOM de las páginas webs que se visitan. Las características de este archivo son:

- Ejecución en entorno aislado para evitar conflictos con el código JavaScript de la página.
- Capacidad de lectura y modificación del contenido de las páginas web.
- Comunicación bidireccional entre los distintos componentes de la extensión.
- Inyección dinámica o estática dependiendo de las necesidades que tenga en las distintas páginas web.

Para instalar extensiones de Google Chrome de manera oficial hay que hacerlo desde el Chrome Web Store en el cual se implementa un proceso de revisión que evalúa todas

las extensiones según criterios estrictos antes de que estén disponibles para el uso público. [17] Estas:

- Realizan una revisión automatizada de manera inicial para detectar cualquier tipo de software malicioso.
- Realizan una revisión manual en el caso de que los permisos que solicite la extensión sean sensibles.
- Exigen políticas de privacidad obligatorias a todas las extensiones que manejen información personal de los usuarios.
- Permiten que los desarrolladores a los cuales se les ha rechazado su extensión apelen para que se revise su caso.

Además, las extensiones tienen que cumplir las políticas de privacidad y de seguridad de Google las cuales exponen: que únicamente se pueden solicitar los permisos que sean estrictamente necesarios para el buen funcionamiento, la trasparencia de datos dejando claro que datos se recopilan y como se utilizan estos datos, se prohíbe que se le proporcionen los datos a brokers o revendedores y se limita la modificación de las páginas web sin consentimiento explícito.

Actualmente existen varias extensiones que se encargan de controlar el contenido, algunas de ellas son bloqueadores de anuncios, filtros de contenido para adultos, herramientas de bienestar digital controlando el tiempo en pantalla o el tiempo en determinadas páginas web o detectores de noticias falsas.

A pesar de todas las nuevas actualizaciones existen varias limitaciones. Existen problemas con:

- La latencia cuando se procesa el contenido dinámico.
- Las limitaciones de API para acceso a ciertos tipos de datos con la nueva versión 3.
- Dependencia de conectividad para verificación en tiempo real.
- Evasión mediante JavaScript complejo en sitios modernos.

2.4 Servidor de Clasificación "ProAnaMia"

El servidor proporcionado por el tutor del TFG implementa un sistema complejo de clasificación de texto basado en técnicas avanzadas de procesamiento de lenguaje natural y aprendizaje automático supervisado.

Realiza un procesamiento lingüístico avanzado. Con este, hace un análisis morfosintáctico usando Stanza, limpieza del contenido web mediante BeautifulSoup para extraer el texto HTML, normalización lingüística eliminando las tildes, signos de puntuación, etc. También elimina los stop words del castellano y reduce las palabras a su forma canónica.

Para entrenar a los modelos se recopilaron 3,5 millones de documentos de las 9 categorías en las cuales el servidor va a clasificar la información de las webs. El peso total de los archivos de entreno es de 454Gb que es en muchos casos la mitad o todo el

almacenamiento disponible de la mayor parte de los ordenadores hoy en día. Para evitar duplicados con diferentes nombres se realizó un filtrado del conjunto original, utilizando la firma "hash" para reducir el tamaño de las muestras, y de esta manera poder comprobar de manera más sencilla si los textos eran redundantes. Gracias a esto se pudo pasar de 3,5 millones de documentos a 491.000 muestras finales. Luego se utilizó TF-IDF transformando los textos en vectores numéricos para calcular la importancia relativa de cada término y mejorar las clasificaciones.

Este servidor utiliza 3 modelos que han sido entrenados para categorizar la información y proporcionar un porcentaje con la fiabilidad de la asignación a la categoría:

- **K-Nearest Neighbors**: es un algoritmo que clasifica basándose en la similitud con los "k" vecinos más cercanos (13 el mejor número de vecinos). Es efectivo para patrones locales y decisiones interpretables, pero, es computacionalmente intensivo para datesheets grandes. Este ha conseguido una precisión del 98,96% por lo que se posiciona en el primer lugar en cuanto a eficacia.
- **Multinomial Naive Bayes**: es un clasificador lingüístico basado en el teorema de Bayes. El parámetro de suavizado con mejores resultados α es de 0,9. A pesar de esto, este algoritmo es el que ha obtenido peor puntuación con un 74,01% de efectividad.
- **Multinomial Logistic Regresion**: es un algoritmo que consiste en un modelo lineal con regularización para problemas multiclase. Este método ha obtenido un 94,4% de efectividad.

Con el servidor, se hace la predicción para saber en cuál de las 9 categorías está el texto y con eso se podrá realizar el filtro de contenido de la extensión.

Gracias a que el servidor está diseñado como un servicio HTTP el procesamiento es en tiempo real. Además, se puede ir actualizando los modelos reentrenándolos con nuevos datos actualizados para mejorar el resultado.

3 TECNOLOGÍAS

En este capítulo se expondrán las tecnologías que fueron usadas en la realización de este TFG.

3.1 VISUAL STUDIO CODE

Visual Studio Code es un editor de código fuente desarrollado por Microsoft que se ha convertido en una de las herramientas preferidas de millones de programadores en todo el mundo. Es de código abierto y está disponible tanto en Windows como en Linux y macOS.

Una de sus ventajas es que funciona con múltiples lenguajes, aunque su soporte nativo es para JavaScript, TypeScript y Node.js, también se utiliza para programar con otros lenguajes como Python, C++, C#, Java, etc. Además, es ligero y rápido ya que, a diferencia de otros IDEs, busca encontrar un equilibrio entre la eficiencia y la agilidad, por lo que la edición de código es muy fluida. Una de sus mejores características es que es extensible. Contiene un Marketplace de extensiones que nos ayudan a añadir funcionalidades, soporte, herramientas de depuración, integración con otras plataformas y muchas más cosas de manera sencilla, por lo que termina estando muy personalizado.

Se ha elegido este editor por varias razones:

- La interfaz de usuario es completa y fácil de usar. Además, cuenta con funcionalidades como el resaltado de la sintaxis y el autocompletado inteligente que a la hora de realizar el código fueron de gran ayuda.

- Cuenta con soporte nativo integrado para tecnologías web (HTML, CSS, JavaScript y JSON) que son usadas para poder realizar la extensión.
- Es capaz de gestionar proyectos de varios archivos de distintas tecnologías. El proyecto necesita que se pueda gestionar el código de Python del servidor, los archivos HTML, CSS y JavaScript de la extensión, etc. Con VS Code se puede abrir varios archivos usando su explorador de archivos sin importar la tecnología que se esté usando y de esta manera poder tener una vista global del proyecto.

3.2 GITHUB

GitHub es una plataforma que nos permite almacenar, compartir y colaborar con código en la nube. Utiliza el sistema de control de versiones Git el cual ayuda a mantener un historial de todos los cambios que se han ido haciendo, detallando el cambio, quién los hace y en que momento, a parte de permitirnos el volver a código antiguo si fuera necesario. Se ha utilizado en este proyecto sobre todo para futuras mejoras y colaboraciones. Al dejar el proyecto en abierto, queda disponible la colaboración con otros desarrolladores en el caso de que quisieran mejorar tanto el servidor como la extensión.

3.3 PYTHON

Python es un lenguaje de programación que esta experimentando un auge importante en su frecuencia de uso por parte de los programadores gracias a su sintaxis simple y clara, las cuales hacen más fácil su aprendizaje y lectura. Este lenguaje fue usado para desarrollar y actualizar el servidor “ProAnaMia”. Se escogió este lenguaje por sus librerías especializadas que necesita usar el servidor como por ejemplo la NLTK para procesamiento de texto, Stanza para el análisis de texto avanzado, etc. Además, se utilizó la clase BaseHTTPRequestHandler que crea un servidor HTTP que procesa las peticiones y devuelve en JSON las clasificaciones. Por último, se usó pyhton por la necesidad de cargar los modelos de machine learning que están almacenados en formato pickle, y con el comando de Python pickle.load() se fue capaz de utilizar estos modelos sin necesidad de hacer tareas más complejas.

3.4 CHROME DEVTOOLS

Google Chrome DevTools son las herramientas de desarrolladores de Chrome. Son un grupo de utilidades integradas en la aplicación de Google Chrome que ayudan a los desarrolladores y diseñadores web a inspeccionar, depurar y modificar el código de una página web en tiempo real.
Se utilizó para comprobar el funcionamiento de la extensión tanto detectando errores que VS Code no era capaz de detectar, como para ver el contenido del almacenamiento local de la extensión para ver la lista negra de URLs y el feedback de los usuarios.

3.5 JSON

JSON es un formato ligero que está diseñado para compartir datos de manera sencilla. Su sintaxis es fácil de leer y escribir tanto para humanos como para aplicaciones. Tiene menor volumen de datos en comparación con XML. El manifest.json es el archivo para el cual se usó esta tecnología. Gracias a esta se pudo definir los permisos para el almacenamiento, para visualizar el texto de las ventanas y para abrir una nueva ventana. También ha servido para realizar la comunicación cliente-servidor ya que el cliente envía un POST con el texto en claro al servidor y este último devuelve la respuesta con las predicciones de categoría y probabilidad en JSON. Además, se utiliza esta tecnología para almacenar tanto el feedback como la lista negra de webs en el almacenamiento local de la extensión.

3.6 HTML

HTML es el lenguaje de programación estándar para contenido en la web. No es un lenguaje de programación tradicional, sino un lenguaje de marcado que define la semántica y la organización de los elementos de una página web. Se podría decir que es parecido al esqueleto de la extensión en este caso. Con HTML se han realizado tanto el popup.html como el blocked.html. El primer archivo es la interfaz de la extensión y el segundo es una web que realiza la función de informar sobre el bloqueo al usuario dándole la opción de regresar al buscador de Google.

3.7 CSS

CSS es un lenguaje de hojas de estilo el cual sirve para diseñar y dar estilo a las webs. Con CSS conseguimos ver de una manera visual los documentos escritos en HTML o en los que estan basados en XML. En si mismo, no es un lenguaje de programación, es un conjunto de reglas que marcan a los navegadores como deben mostrarse los distintos elementos de las páginas web. Se ha utilizado CSS para garantizar que tanto la interfaz de la extensión como la página web de bloqueo se vean de manera correcta independientemente del tamaño de la ventana que se utilice. Además, se añadieron mejoras a la experiencia del usuario como los botones o el feedback visual.

3.8 JAVASCRIPT

JavaScript es un lenguaje de programación complejo y completo orientado a objetos y es usado de manera habitual para el desarrollo de páginas web. Es esencial para poder

programar aplicaciones y contenido dinámico en la web y se ejecuta en el navegador haciendo posible que los desarrolladores creen webs interactivas y aplicaciones en línea. En la extensión, el lenguaje principal es JavaScript, ha sido utilizado tanto para el background.js como para el content.js y el popup.js. En el primero implementa la lógica principal de la extensión. Se encarga de la comunicación con el servidor, la gestión de la lista negra, el manejo de las reglas para redirigir a la página de bloqueo… El segundo, ejecuta el MutationObserver para poder monitorear los cambios dinámicos del DOM que de otra manera sería mucho más complicado de gestionar y extrae el contenido textual para enviárselo al background. El último, controla la interfaz de usuario. Se encarga de actualizar la lista negra en el popup y gestiona el sistema de feedback del usuario.

Gracias a usar JavaScript la extensión funciona únicamente el lado del cliente con comunicación al servidor únicamente para el análisis de contenido y de esta manera se consigue un mejor rendimiento y experiencia de usuario

4 DESARROLLO PRÁCTICO

En esta parte del trabajo se exponen los aspectos técnicos empleados en el desarrollo de la extensión de Chrome para la detección y bloqueo de contenidos “ProAnaMia”. Se detallará el funcionamiento y la implementación del servidor de clasificación y posteriormente se describirá el desarrollo de cada componente de la extensión.

4.1 EL SERVIDOR

Para que el servidor funcione de manera correcta lo primero que se tiene que hacer es tener todos los archivos en las ubicaciones correctas. Debe de estar el archivo de Python en la carpeta principal junto con otras dos carpetas: data, que contiene una carpeta models con los modelos de machine learning que necesita el servidor y otra carpeta pickles con el vectorizador TF_IDF, y otra carpeta venv con el entorno virtual.

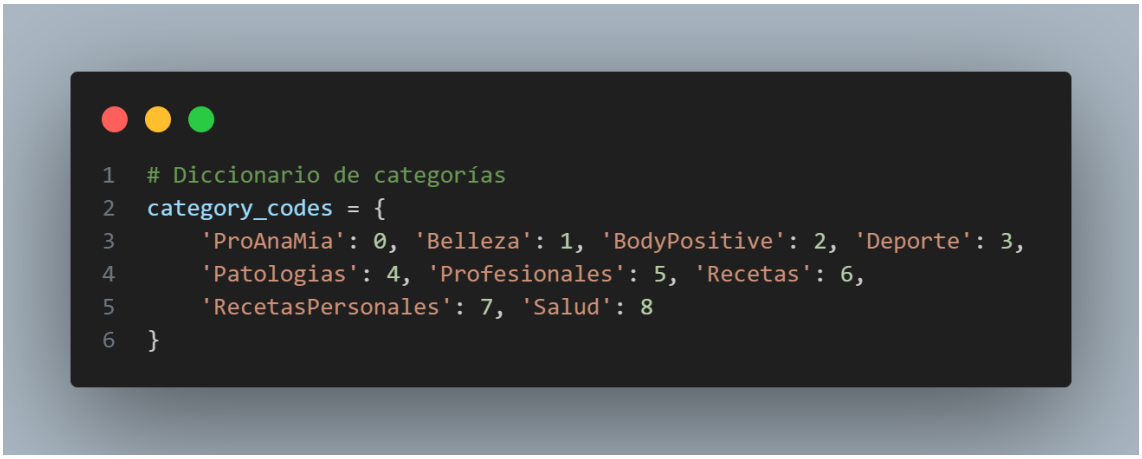
Después de esto, se deben instalar todas las dependencias:

- pip install --upgrade pip (para actualizar el pip)
- pip install nltk
- pip install stanza
- pip install scikit-learn
- pip install beautifulsoup4
- pip install lxml
- pip install numpy
- pip install pandas

Las librerías nktl y stanza se instalan para que pueda analizar lingüísticamente, scikit-learn para la clasificación que hace en categorías, el servidor web para recibir peticiones HTTP y el BeautifulSoup para que podamos extraer el texto limpio.

Como se puede ver en la Ilustración 2, el servidor va a categorizar el contenido de las páginas web en 9 grupos diferentes. El más importante es el código de categoría 0 que es el de "ProAnaMia". Ya que los demás son contenidos relacionados con:

1. Cosméticos o estética
2. Aceptación corporal positiva
3. Deportes o fitnes
4. Medicina sobre enfermedades
5. Profesionales de la salud
6. Recetas de cocina
7. Contenido general de salud y bienestar



```
# Diccionario de categorías
category_codes = {
    'ProAnaMia': 0, 'Belleza': 1, 'BodyPositive': 2, 'Deporte': 3,
    'Patologias': 4, 'Profesionales': 5, 'Recetas': 6,
    'RecetasPersonales': 7, 'Salud': 8
}
```

Ilustración 2 Categorías Servidor

Luego, después de descargar los recursos de procesamiento de texto en español e iniciar los pipelines de análisis linguistico junto con las herramientas de limpieza de texto (quitar signos, etc), se cargan los modelos de ia entrenados: KNN, MNBC y LRC. (Ilustración 3)

```python
# Modelo KNN
with open(os.path.join(path_models, 'best_knnc.pickle'), 'rb') as data:
    knn_model = pickle.load(data)
# Modelo MNBC
with open(os.path.join(path_models, 'best_mnbc.pickle'), 'rb') as data:
    mnbc_model = pickle.load(data)
# Modelo LRC
with open(os.path.join(path_models, 'best_lrc.pickle'), 'rb') as data:
    lrc_model = pickle.load(data)

print("¡Servidor listo! Modelos cargados y listos para predecir.")
```

Ilustración 3 Carga de los modelos

Una vez ya cargados los modelos el servidor y de que se prepare el texto para que los modelos de Machine Learning puedan procesarlos, se pasa a realizar las predicciones con los modelos de manera independiente para luego calcular la probabilidad (predict_from_model()) y la categoría. Se devuelve el resultado como un porcentaje ya que de esta manera podemos poner el filtro con un umbral mínimo antes de bloquear el contenido. En el caso de la extensión, el umbral mínimo de contenido "ProAnaMia" ha sido del 5%, aunque en caso de que se quiera editar, únicamente hay que cambiar el valor en el archivo background.js de la extensión en las primeras líneas en "CATEGORY_0_THRESHOLD".

Por último, tiene la clase Servidor, que recibe el texto de la extensión, lo procesa, lo analiza con los 3 modelos y devuelve las predicciones en formato JSON.

Cuando se arranca el sevidor, primero se realizarán las siguientes operaciones:

- Descargar los recursos NLTK y Stanza
- Iniciar los pipelines de NLP
- Carga de los modelos de Machine Learning
- Inicio del servidor

Después de esto, el servidor se ejecuta en el puerto 80(aunque es posible cambiar el puerto en el archivo de python del servidor en las últimas líneas).

Una vez hecho todo esto, el funcionamiento del servidor consiste en que este cuando recibe un POST:

- Extrae el texto en plano usando BeautifulSoup para limpiar el HTML
- Elimina los caracteres especiales y normalización
- Tokemiza y lematiza usando WordNetLemmatizer
- Elimina los stop words del castellano

- Vectoriza TF-IDF para convertir el texto en características numéricas
- Predice con los tres modelos de manera independiente
- Devuelve los resultados de las predicciones en formato JSON

En la respuesta JSON podremos ver la categoría de la predicción con el porcentaje de esta y si cualquiera de los 3 modelos detecta que el contenido es “ProAnaMia” con un 5% o más el contenido será marcado como peligroso para la extensión.

Uno de los requisitos de funcionamiento de la extensión es que los tiempos de respuesta debían de ser muy rápidos para no entorpecer la experiencia del usuario y gracias a la arquitectura HTTP integrada en el servidor y a que los modelos se cargan una única vez durante el arranque, lo que hace que las predicciones posteriores sean muy eficientes en el uso de los recursos,el tiempo de procesamiento es inferior al segundo.

El servidor se integra con la extension de la siguiente manera:

1. Después de que el content.js haya extraido todo el texto de las páginas web, lo envía al background.js que es el encargado de enviarselo mendiante un POST de HTTP al servidor.
2. El servidor envía el resultado al background con el formato JSON y el background comprueba si se cumple el umbral mínimo el cual se ha fijado en un 5% para saber si tiene que bloquear el contenido o no.

4.2 LA EXTENSIÓN

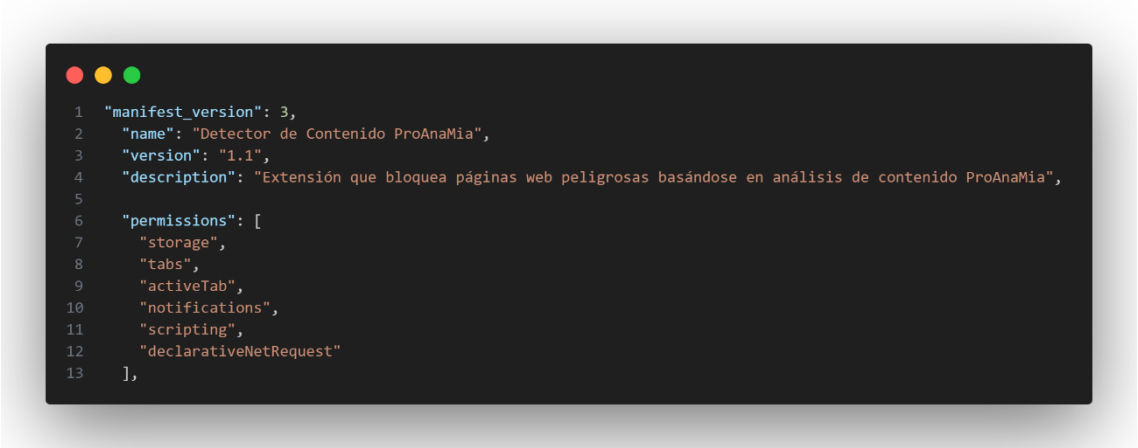
Para desarrollar la extensión hace falta crear los siguientes archivos: manifest.json, background.js, content.js, popup.js, blocked.html y popup.html.

4.2.1 Manifest.json

El archivo manifest contiene la configuración esencial de la extensión de Google Chrome. En él, se tiene que definir qué permisos tiene y como interactúa esta con el navegador.

Lo primero que aparece en este archivo es el nombre y una pequeña descripción de lo que hace esta extensión para que tanto Google como los usuarios que la descarguen sepan que hace.

Luego, se solicitan todos los permisos que necesita la extensión para funcionar de manera correcta, como por ejemplo permisos para: guardar URLs de manera local, redirigir pestañas a la pagina de bloqueo, leer el contenido de la pestaña actual, sistema de bloqueo automático, etc. (Ilustración 4)



```
"manifest_version": 3,
  "name": "Detector de Contenido ProAnaMia",
  "version": "1.1",
  "description": "Extensión que bloquea páginas web peligrosas basándose en análisis de contenido ProAnaMia",

  "permissions": [
    "storage",
    "tabs",
    "activeTab",
    "notifications",
    "scripting",
    "declarativeNetRequest"
  ],
```

Ilustración 4 Descripción y permisos de la extensión

A continuación, se ponen las componentes de la extensión: background, content_scripts, action y web_accessible_resources. Y además, se añade el icono de la extensión con diferentes proporciones.

4.2.2 Background.js

El background es el cerebro de la extensión. Se encarga de coordinar la lógica de análisis, bloqueo y gestión de URLs peligrosas según el contenido “ProAnaMia”.

Lo primero que se hace en este archivo es fijar la dirección del servidor que en mi caso es el puerto 80 y el umbral mínimo para el cual la extensión bloquea el contenido que se ha puesto en el 5%.

Luego se crea la lista de URLs que están bloqueadas para que tengamos esta lista negra de páginas con contenido peligroso.

Con Chrome.runtime.onInstalled.addListener se ha recuperado las webs bloqueadas que están guardadas en el almacenamiento local, se inicia la lista en la memoria y se sincroniza los datos entre diferentes sesiones.

```
chrome.runtime.onMessage.addListener((msg, sender, sendResponse) => {
  if (msg.type === 'ANALYZE_PAGE') {
    handleAnalyzePage(msg.content, sender.tab.id, sender.tab.url)
      .then(result => sendResponse(result))
      .catch(() => sendResponse({ blocked: false }));
    return true;  // Mantener canal abierto
  }

  if (msg.type === 'GET_BLOCKED_URLS' || msg.type === 'GET_BLOCKED_DOMAINS') {
    sendResponse({ domains: Array.from(blockedUrls) });
  }

  if (msg.type === 'SUBMIT_FEEDBACK') {
    saveFeedback(msg.feedback).then(() => sendResponse({ success: true }));
    return true;
  }
});
```

Ilustración 5 Gestión de mensajes

En la Ilustración 5 se puede ver cómo se gestionan los mensajes en la extensión. Definimos 3 tipos de mensajes: Analyze que se encarga de analizar los contenidos de las páginas web, Get_bloqued_urls que obtiene la lista de las páginas bloqueadas y Submit_feedback que guarda en el almacenamiento local el feedback del usuario.

- HandleAnalyzaPage():

Se encarga del análisis de las páginas web. Primero si la web ya se encuentra en la lista negra de URLs bloqueadas, directamente te bloquea el acceso y te redirecciona a la página de bloqueo. Luego, si la web no se encuentra en la "black list", analiza el contenido enviándoselo al servidor y luego reacciona bloqueando la página si el contenido es identificado como peligroso.

- AnalyzeContent():

Esta función se comunica con el servidor de python para analizar el texto de las webs.

Empieza enviando al servidor el contenido de las páginas web y luego recibe las predicciones de los 3 modelos de IA y evalúa si alguno de los modelos detecta peligro. Está diseñado para que, aunque no todos los modelos detecten el contenido como "ProAnaMia" se bloquee igualmente. Con que solo uno o más de los modelos de positivo en esta categoría, la extensión lo detecta como peligro.

```
async function analyzeContent(text) {
  try {
    const resp = await fetch(SERVER_URL, {
      method: 'POST',
      headers: { 'Content-Type': 'text/plain' },
      body: text
    });
    if (!resp.ok) throw new Error(`HTTP ${resp.status}`);
    const data = await resp.json();
    const isKnn = checkDanger(data.knn_category, data.knn_proba);
    const isMnbc = checkDanger(data.mnbc_category, data.mnbc_proba);
    const isLrc = checkDanger(data.lrc_category, data.lrc_proba);
    return { isDangerous: isKnn || isMnbc || isLrc };
  } catch (error) {
    console.error('[background] Error en análisis de contenido:', error);
    return { isDangerous: false };
  }
}
```

Ilustración 6 AnalyzeContent

- CheckDanger():

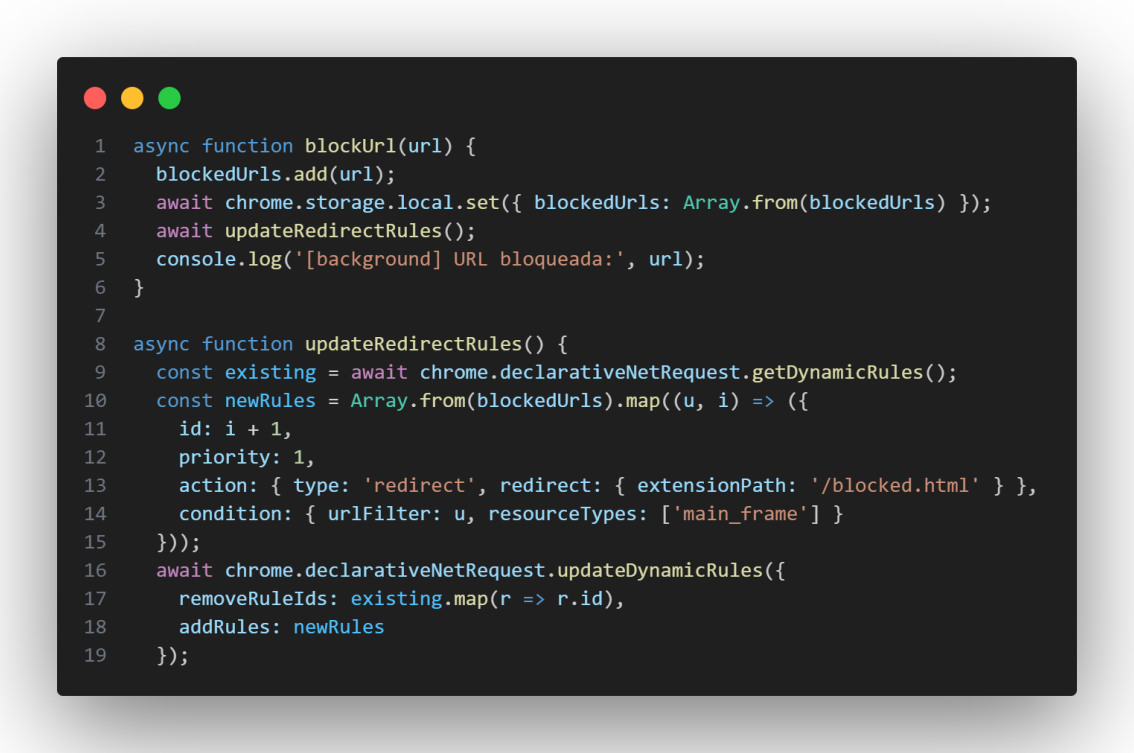
Determina cuando una predicción indica contenido peligroso o no. Esto lo hace comprobando que la categoría sea la de “ProAnaMia” y que el umbral de probabilidad sea mayor del 5% para evitar que haya falsos positivos por predicciones de baja confianza.

- BlockUrl():

Añade las páginas que se han determinado como peligrosas a la lista de bloqueo para optimizar tiempos de funcionamiento y además tener una lista con todas las páginas peligrosas por si se quieren realizar más funciones. Luego, actualiza las reglas de redirección.

- updateRedirectRules():

Utiliza la API de Chrome, declarativeNetRequest, para crear reglas que redirigen a las páginas bloqueadas. Además, las va actualizando y reemplazando con las nuevas actualizaciones.



```
async function blockUrl(url) {
  blockedUrls.add(url);
  await chrome.storage.local.set({ blockedUrls: Array.from(blockedUrls) });
  await updateRedirectRules();
  console.log('[background] URL bloqueada:', url);
}

async function updateRedirectRules() {
  const existing = await chrome.declarativeNetRequest.getDynamicRules();
  const newRules = Array.from(blockedUrls).map((u, i) => ({
    id: i + 1,
    priority: 1,
    action: { type: 'redirect', redirect: { extensionPath: '/blocked.html' } },
    condition: { urlFilter: u, resourceTypes: ['main_frame'] }
  }));
  await chrome.declarativeNetRequest.updateDynamicRules({
    removeRuleIds: existing.map(r => r.id),
    addRules: newRules
  });
```

Ilustración 7 BlockUrl y updateRedirectRules

- redirectToBlocked():

Función que redirecciona a la página de bloqueo en el momento que se determina que la URL es peligrosa.

- saveFeedback():

Se encarga de enviar y guardar en el almacenamiento local el feedback que los usuarios ponen en la extensión.

4.2.3 Content.js

El archivo content se encarga de extraer el texto de las páginas web e iniciar la detección de contenido peligroso. Extrae el texto limpio de las webs y envía ese texto a background para que sea analizado con los modelos ML. Además, monitorea los posibles cambios dinámicos de las webs, ya que existen páginas como las SPA modernas que cargan nuevo contenido sin necesidad de recargar.

- extractPageContent():

Para realizar esto, lo primero que se hace es extraer el texto limpio de la página web. Para esto, clonamos el “body” para no cambiar la página que el usuario ve, elimina todas las etiquetas que

son irrelevantes para el análisis de contenido peligroso, toma el texto visible y añade al texto extraído el título de la página. (Figura 8)

- analizePage():

Llama a la extracción de texto y lo analiza. Para esto último, envía el texto al "background" y si la página es bloqueada avisa por consola. (Ilustración 8)

```
function extractPageContent() {
  const clone = document.body.cloneNode(true);
  clone.querySelectorAll('script, style, noscript').forEach(el => el.remove());
  const text = clone.innerText.trim();
  const title = document.title.trim();
  return `${title}\n${text}`.slice(0, 10000);
}

// Envía contenido al background
async function analyzePage() {
  const content = extractPageContent();
  if (content.length < 50) return;

  chrome.runtime.sendMessage({ type: 'ANALYZE_PAGE', content }, response => {
    if (response.blocked) {
      console.log('[content] Página bloqueada');
    }
  });
}
```

Ilustración 8 Extracción de contenido y envío al background

Después de esto, crea un multiobserver que vigila cuando cambia el DOM (el contenido de la página cambia). Comprueba después de un timer si la página a cambiado, aunque no se haya recargado.

Por último, está el inicio automático. Cuando la web termina de cargar el DOM, comienza a observar cambios y realiza el primer análisis. Una vez termina de cargar todo, vuelve a lanzar otro análisis extra para asegurar cobertura.

4.2.4 Popup.js

Este archivo es el encargado de manejar la interfaz de usuario emergente de la extensión. Esta solo aparece cuando el usuario hace clic en el icono de la extensión en Chrome y su función principal es mostrar los dominios bloqueados y permitir que el usuario envíe feedback de manera sencilla desde esta ventana fácil de usar.

- loadBlockedDomains():

Esta función extrae la lista de URLs bloqueadas del archivo background. Y en caso de que no lo consiga devuelve un error.

- updateBlockedDomainsUI():

Actualiza la interfaz visual con los dominios que están bloqueados. Además, crea elementos HTML para cada URL.

- handleFeedbackSubmit():

Después de configurar los listeners, esta función maneja el envío del feedback desde el usuario al admin de la extensión. Define como se envía el feedback obligando a completar como mínimo el apartado de mensaje para poder enviarlo. También permite al usuario especificar el tipo de feedback que va a mandar como por ejemplo un falso positivo, falso negativo, un error técnico o una sugerencia. Todo esto, puede estar acompañado con la URL de la que se esta reportando en otro recuadro a rellenar para que los admins puedan revisarlo de manera más sencilla.

```
async function handleFeedbackSubmit(e) {
    e.preventDefault();

    const formData = new FormData(e.target);
    const feedback = {
        type: formData.get('feedback-type'),
        message: formData.get('feedback-message'),
        url: formData.get('feedback-url') || ''
    };

    if (!feedback.message.trim()) {
        showStatus('Por favor ingresa un mensaje', 'error');
        return;
    }

    try {
        showStatus('Enviando feedback...', 'info');

        const response = await chrome.runtime.sendMessage({
            type: 'SUBMIT_FEEDBACK',
            feedback: feedback
        });

        if (response.success) {
            showStatus('Feedback enviado correctamente', 'success');
            e.target.reset(); // Limpiar formulario
        } else {
            showStatus('Error enviando feedback', 'error');
        }
    } catch (error) {
        console.error('Error:', error);
        showStatus('Error enviando feedback', 'error');
    }
}
```

Ilustración 9 Manejo del feedback

4.2.5 Blocked.html

Blocked.html es una página de advertencia con la cual se avisa al usuario que está intentando ver contenido “ProAnaMia” y con ella se evita que se pueda ver ese contenido.

Esta página aparece en dos ocasiones. La primera es cuando el usuario intenta acceder a páginas que ya se encuentran en la lista de URLs bloqueadas ya que se han analizado con anterioridad y el resultado fue que la web contaba con contenido peligroso. Y la segunda es cuando se detecta que la página es maliciosa en tiempo real con el content.js extrayendo el contenido sobre la marcha y enviándoselo al background.js. Este último lo analiza y si es peligroso se redirige inmediatamente a esta página de bloqueo.

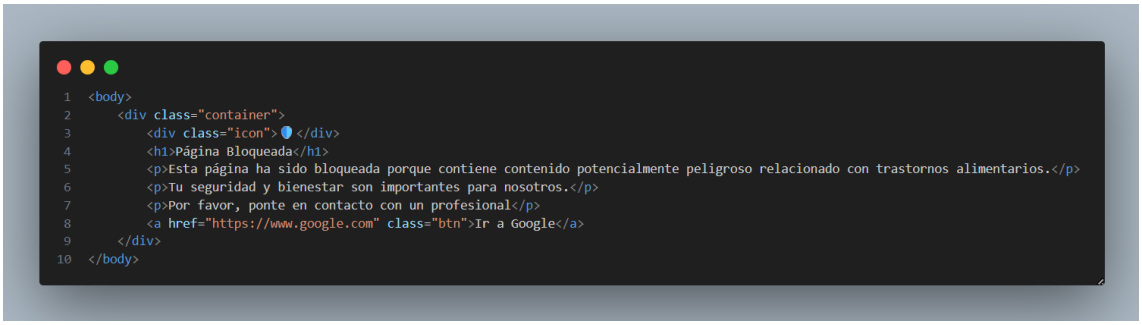


Ilustración 10 Body de Blocked.html

En el body de este archivo se explica la razón por la cual se bloqueo la pagina a la cual se intentaba acceder y se recomienda al usuario que se ponga en contacto con un profesional para evitar más daños.

También se ha añadido un botón de "Ir a Google" que redirige al usuario a Google.com para que pueda seguir navegando en la misma ventana de manera sencilla para que no tenga que abrir una nueva o editar la URL de la ventana.

La demás parte del archivo es código html que define la estética del fondo y como se distribuye todo en la pagina web de manera visual, como por ejemplo el color de letra y de fondo, el tipo de letra, etc.

4.2.6 Popup.html

Con popup.html definimos la interfaz gráfica emergente que permite al usuario visualizar las URLs bloqueadas y comunicar feedback cuando se clica en el icono de la extensión (icono48.png diseñado exclusivamente para esta extensión). Se ha diseñado de manera que el popup no sea muy invasivo por lo cual el tamaño de este es reducido pero suficiente para que la experiencia del usuario sea satisfactoria.

Lo primero con lo que nos encontramos al clicar en la extensión es con un título que nos describe lo que hace. Luego, el popup esta dividido en 3 secciones, la primera es el apartado de estadísticas en el cual se hace un recuento de las páginas bloqueadas y se le confirma al usuario que la protección se encuentra activa. La segunda es la parte de las URLs bloqueadas. En ella se encuentra de manera detallada todas las webs que se han bloqueado por contener contenido "ProAnaMia" actualizándose en tiempo real. Y la ultima parte es la del feedback, en la cual el usuario podrá enviar todo el feedback que crea necesario tanto para mejorar el funcionamiento actual de la extensión como para proponer mejoras que se puedan implementar en el futuro. Para esto último, puede hacer uso de botones que hacen de manera más sencilla la interacción con el usuario.

```html
<div class="section">
            <h3><span class="icon">💬</span> Reportar Problema</h3>
            <form id="feedback-form" class="feedback-form">
                <div class="form-group">
                    <label class="form-label" for="feedback-type">Tipo de Reporte:</label>
                    <select id="feedback-type" name="feedback-type" class="form-control" required>
                        <option value="">Seleccionar...</option>
                        <option value="false-positive">Falso Positivo</option>
                        <option value="false-negative">Contenido Peligroso No Detectado</option>
                        <option value="bug">Error Técnico</option>
                        <option value="suggestion">Sugerencia</option>
                    </select>
                </div>
                <div class="form-group">
                    <label class="form-label" for="feedback-url">URL (Opcional):</label>
                    <input type="url" id="feedback-url" name="feedback-url" class="form-control"
                           placeholder="https://ejemplo.com">
                </div>
                <div class="form-group">
                    <label class="form-label" for="feedback-message">Descripción:</label>
                    <textarea id="feedback-message" name="feedback-message" class="form-control"
                              placeholder="Describe el problema o sugerencia..." required></textarea>
                </div>
                <button type="submit" class="btn btn-primary" style="width: 100%;">
                    📩 Enviar Reporte
                </button>
            </form>
        </div>
```

Ilustración 11 Feedback en el popup

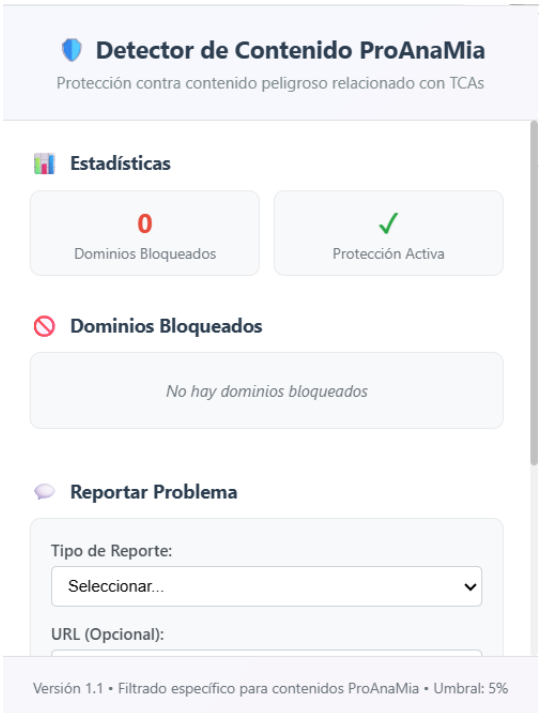


Ilustración 12 Captura de la extensión

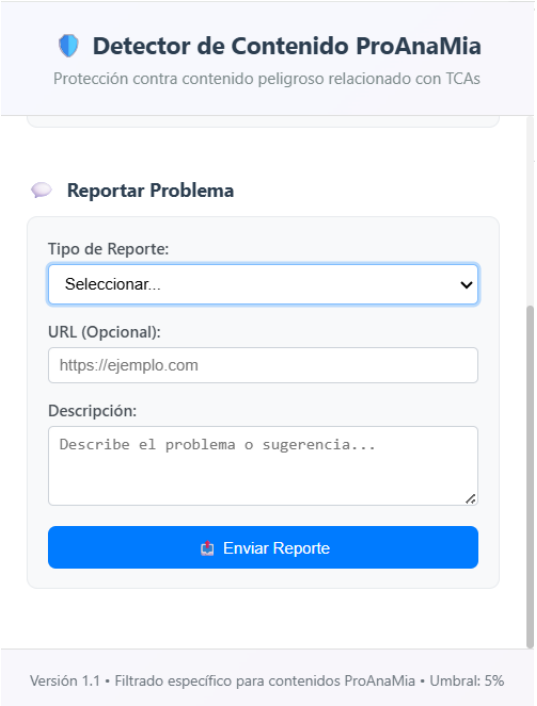


Ilustración 13 Captura del feedback de la extensión

4.3 PROBLEMAS ENCONTRADOS

A continuación, se expondrán una lista con los distintos problemas que han surgido después de realizar las pruebas de funcionamiento durante el desarrollo de este TFG.

4.3.1 Servidor

Uno de los primeros problemas con el que me encontré al comenzar este trabajo fue el de conseguir actualizar el código del servidor para que funcionase despues de las actualizaciones que habían ido teniendo las distintas librerías que se usaron en el momento en el que fue programado por primera vez.

4.3.2 DOM Alterado

Este fue un problema el con el cual me encontré después de haber programado la extensión completa, y al probarla, varias páginas web como X, Google… Salían en el navegador de manera errónea incluso cuando no contenían nada “ProAnaMia”.

Después de muchas pruebas como crear una whitelist para las páginas que daban error para comprobar si el problema era solo con esas o si era otra cosa y de repasar el código de manera exhaustiva, encontré el error en el código del archivo content.js ya que al extraer el contenido se eliminaban los elementos <script>, <style> y <noscript> del DOM para analizarlos y eso provocaba que este quedara de manera parcial sin estilo y scripts. Esto es lo que provocaba que las páginas aparecieran sin formato como se puede apreciar en la Ilustración 13.

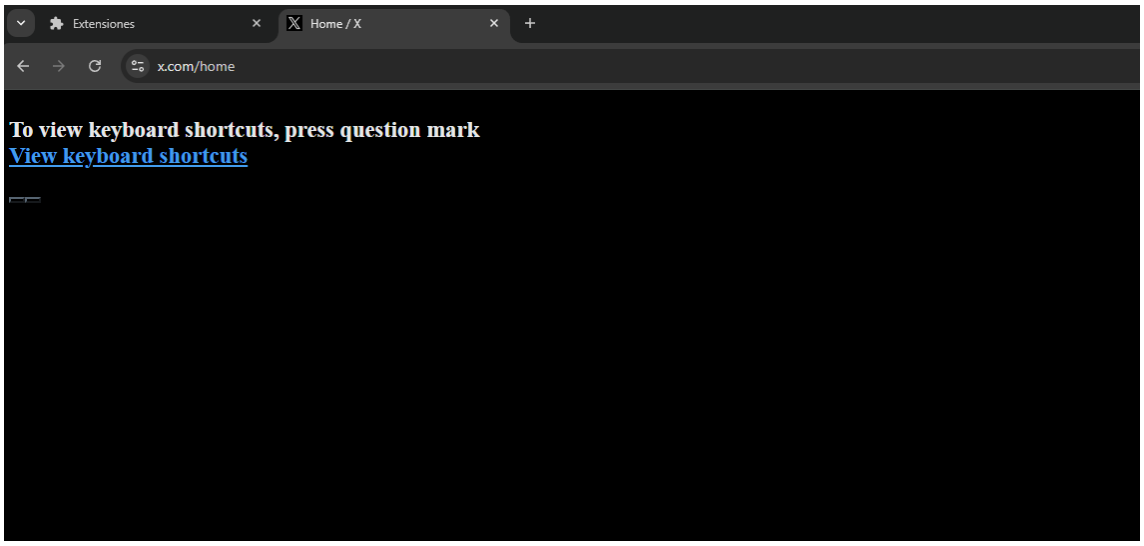


Ilustración 14 Captura error DOM alterado

Para resolver este error, la solución que aplique fue el crear una copia del DOM para que, a la hora de realizar los cambios necesarios y el análisis, no lo hiciera directamente en el DOM original, y de esta manera, el error fue solucionado.

5 RESULTADOS, CONCLUSIONES Y FUTURAS MEJORAS

A continuación, se presentan los resultados obtenidos tras la implementación exitosa del sistema de detección y bloqueo de contenidos “ProAnaMia”. En ella se comprobará el funcionamiento de la extensión frente a distintas situaciones. Después de esto, se exponen las conclusiones de este TFG junto con unas posibles mejoras que se podrán aplicar a futuro para optimizar el funcionamiento de la extensión y el servidor.

5.1 RESULTADOS

Los resultados obtenidos son los que se esperaban conseguir cuando se comenzó este proyecto. La extensión no influye en el uso normal de Google Chrome y solamente actúa cuando detecta el contenido peligroso.

Se ha conseguido implementar la detección de contenido “ProAnaMia” en tiempo real con el bloqueo simultáneo y la redirección a una página de bloqueo si el contenido es peligroso, junto con un sistema de feedback para que la extensión cuente con una mejora continua y una interfaz de usuario sencilla. Todo esto es gracias a que el tiempo de respuesta del servidor es de menos de 1 segundo por análisis. Además, el umbral de

detección es del 5% y la lista negra se almacena en el almacenamiento local por si fuese necesario el consultarla para futuras mejoras.

Lo primero que hay que hacer para que todo funcione de manera correcta es iniciar el servidor y cargar la extensión en el navegador entrando en chrome://extensions/ y adjuntar la carpeta en la que está la extensión.

Una vez cargada la extensión aparecerá el logo de esta en el apartado de extensiones de Chrome y se podrá empezar a hacer las pruebas. (Ilustración 15)

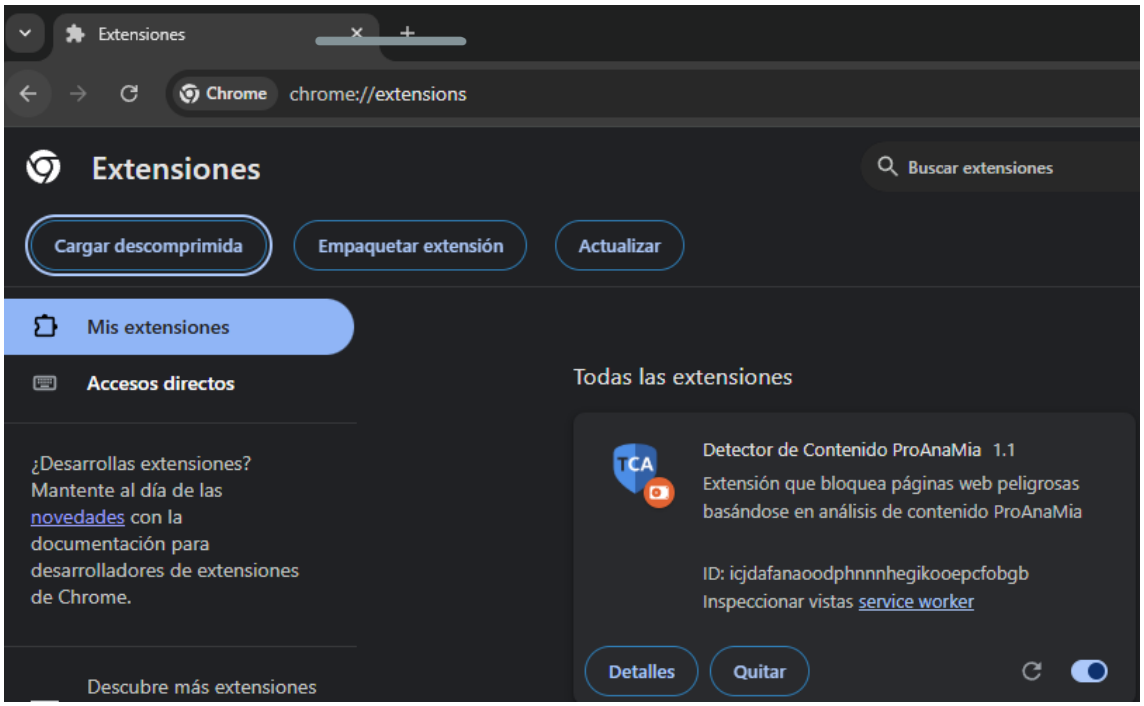


Ilustración 15 chrome://extensions/

Existen 3 opciones cuando navegamos con la extensión. La primera es que intentemos entrar en contenido que es seguro, la segunda es que queramos acceder a contenido dañino por primera vez y la tercera es que queramos acceder a una web la cual se encuentra en la blacklist.

5.1.1 Escenario 1 contenido seguro: marca.com

Al introducir la URL lo primero que se revisa es si esta forma parte de la blacklist, y si no está, la extensión (con content.js) intercepta el contenido para clonarlo, para no alterar el texto original, y lo manda al background.js que mediante un HTTP POST al puerto 80 se lo pasa al servidor de categorización después de que se haya preparado el texto como ya se comentó en apartados anteriores.

El servidor procesa el texto con los 3 modelos de machine learning y calcula las probabilidades de las 9 categorías con cada uno de los modelos de manera independiente y devuelve la respuesta en formato JSON. Como el contenido no es

“ProAnaMia” (Ilustración 16), la página se muestra normal al usuario sin interrupciones y el MutationObserver se queda vigilando a los posibles cambios dinámicos que haya en la web.

```
¡Servidor listo! Modelos cargados y listos para predecir.
Iniciando servidor httpd en el puerto 80...
Predicción con KNN: Categoría=Recetas, Probabilidad=100.00%
Predicción con MNBC: Categoría=Deporte, Probabilidad=16.39%
Predicción con LRC: Categoría=RecetasPersonales, Probabilidad=47.78%
127.0.0.1 - - [01/Sep/2025 12:09:46] "POST / HTTP/1.1" 200 -
```

Ilustración 16 Predicción marca.com

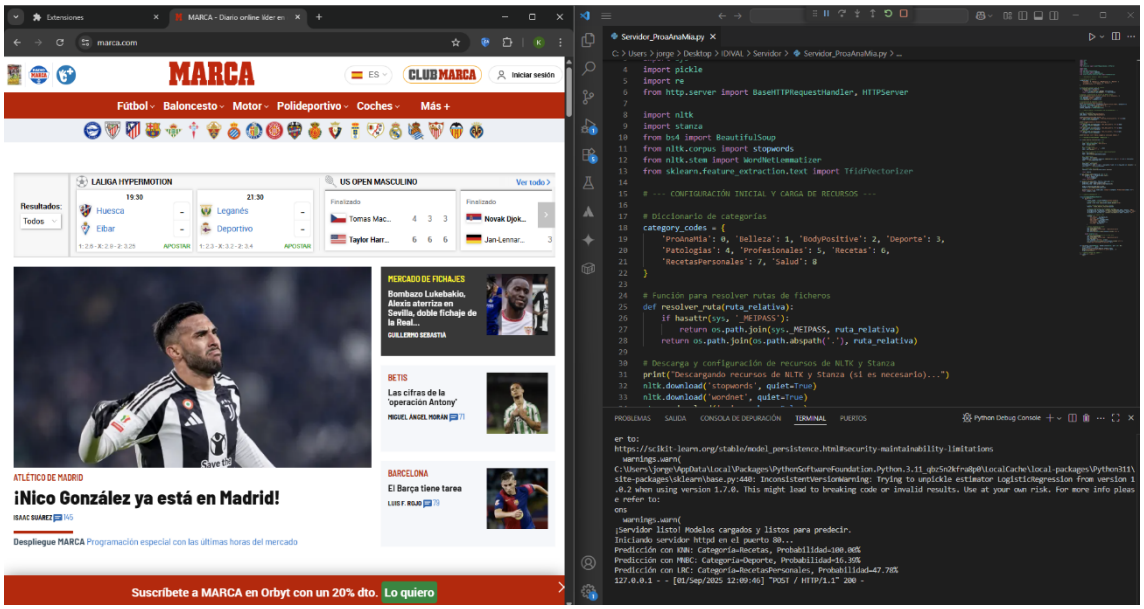


Ilustración 17 Resultado búsqueda marca

5.1.2 Escenario 2 contenido nocivo por primera vez: watpad “ProAnaMia”

Al igual que en el caso anterior, cuando introducimos la URL lo primero que se revisa es si esta forma parte de la blacklist, y si no está, la extensión se lo manda al servidor.

En este caso el servidor al detectar el contenido dañino (Ilustración 19) devuelve el resultado a la extensión que con la función checkDanger() detecta si alguno de los modelos ha clasificado el contenido como “ProAnaMia” y procede a bloquear la página (Ilustración 20). También añade automáticamente la web a la lista negra en el almacenamiento local.

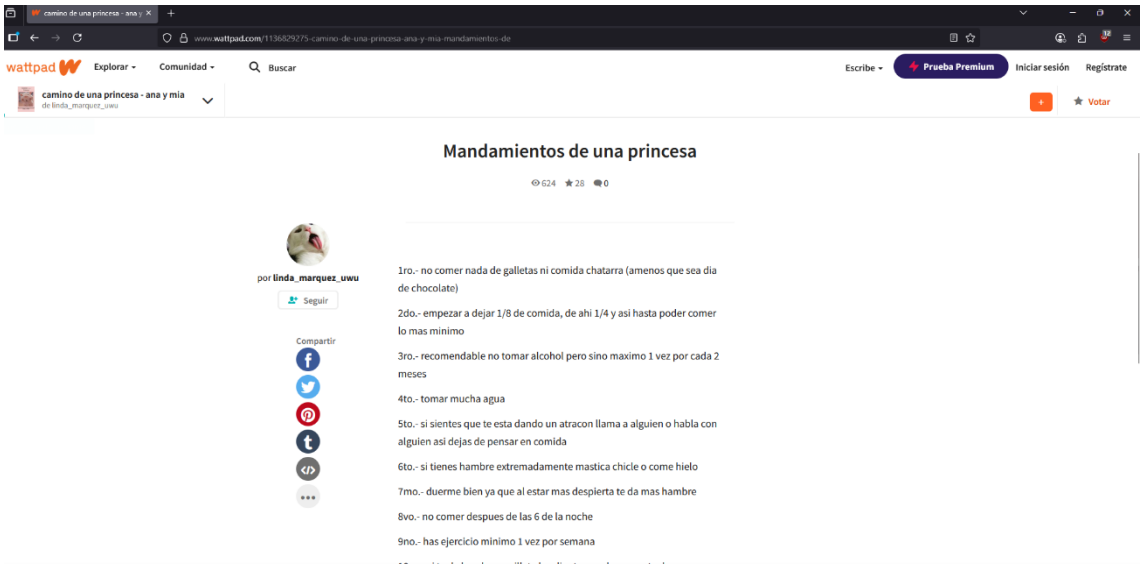


Ilustración 18 Ejemplo de página "ProAnaMia" en otro navegador

```
Predicción con KNN: Categoría=Recetas, Probabilidad=100.00%
Predicción con MNBC: Categoría=ProAnaMia, Probabilidad=30.95%
Predicción con LRC: Categoría=Patologias, Probabilidad=93.73%
127.0.0.1 - - [01/Sep/2025 12:20:43] "POST / HTTP/1.1" 200 -
```

Ilustración 19 Predicción página dañina

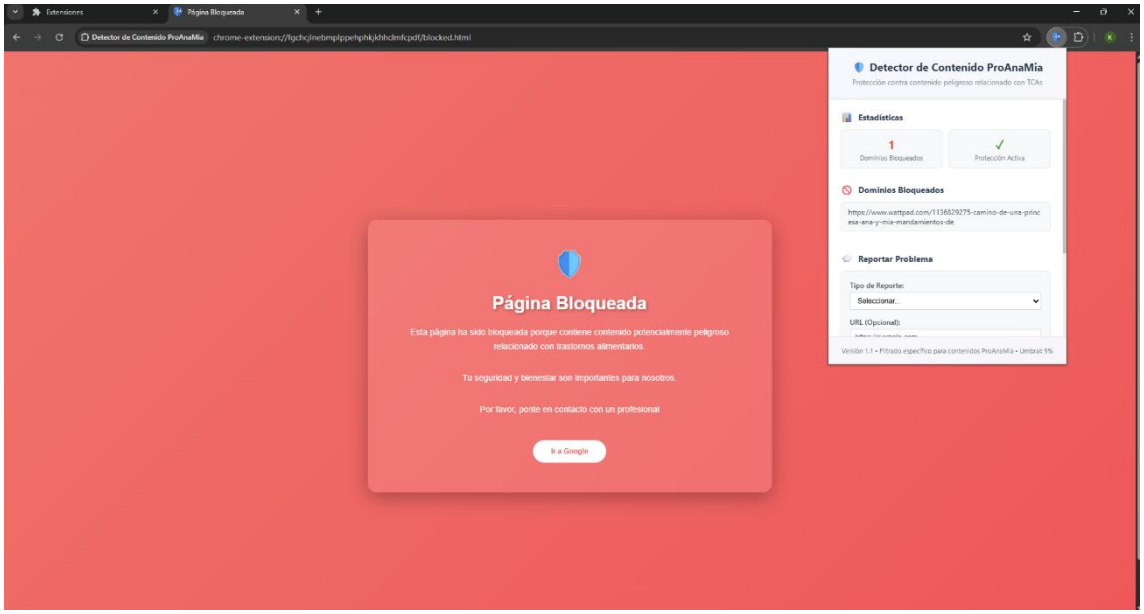


Ilustración 20 Resultado página dañina

5.1.3 Escenario 3 acceso a página contenida en la blacklist: watpad “ProAnaMia”

Como en los casos anteriores, introducimos la URL en el navegador, pero nada más detectar que es una web contenida en la blacklist directamente nos reenvía a la página de bloqueo. (Ilustración 21)

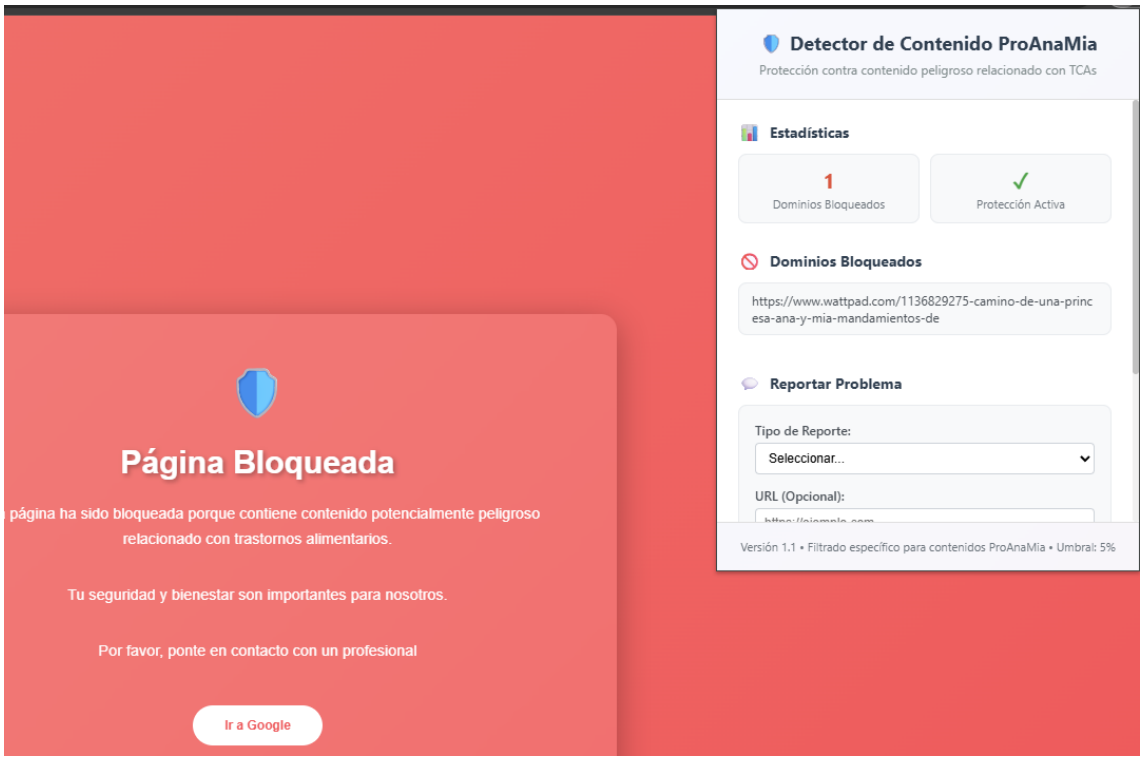


Ilustración 21 Intento de acceso a URL de la blacklist

5.2 CONCLUSIONES

Este trabajo demuestra que es posible el crear un sistema de detección proactiva de contenidos “ProAnaMia” mediante la integración de modelos de machine learning junto con tecnologías web modernas como son las extensiones de Google Chrome.

Una de las ventajas de este proyecto es que cuenta con una arquitectura modular que separa al servidor de clasificación de la extensión de Chrome. Esto ayuda al mantenimiento de ambas partes ya que cada componente es independiente, a que las actualizaciones de los modelos se puedan hacer sin tocar la extensión y, además, se puede implementar en otros navegadores con mínimos cambios.

Al interceptar la información de las páginas web en tiempo real y clasificarla con los modelos, se pone un filtro para que el bloqueo al acceso a páginas con contenidos “ProAnaMia” sea proactivo y esto ayude especialmente a las personas más vulnerables.

El que se haya fijado el umbral del 5% ayuda a evitar falsos positivos priorizando la protección.

Otra de las ventajas de este TFG es que tiene un impacto mínimo en la experiencia de usuario a la hora de usar el navegador ya que el tiempo de respuesta es menor al segundo y la lista negra y el feedback nos ayudan a optimizar las consultas repetidas e implementar mejoras respectivamente.

Este proyecto es muy escalable, ya que a parte de poder añadir al servidor modelos actualizados para la detección de contenido "ProAnaMia" podemos meterle un modelo con nuevas y mejoradas categorías para controlar el acceso a otros contenidos.

Como conclusión, la extensión cumple su fin, controla el contenido "ProAnaMia" de la web. Este proyecto, a diferencia de los métodos tradicionales de filtro, ha conseguido crear un filtro en tiempo real capaz de detectar leguaje críptico que intenta evadir su filtrado y, además, lo hace sin afectar a la experiencia del usuario.

5.3 FUTURAS MEJORAS

Aunque se hayan cumplido todos los objetivos planteados en el trabajo, esta es una herramienta que puede ser mejorada en el futuro para optimizar su funcionamiento y para añadir nuevas funcionalidades.

Se pueden actualizar los modelos para que incluyan nuevo contenido "ProAnaMia" que haya sido identificado y para que se adapte a la evolución del lenguaje de estas subculturas.

A la extensión se le puede implementar un sistema de usuarios para que los padres puedan navegar sin ningún tipo de bloqueo y puedan realizar funciones que no sean posibles sin iniciar sesión con la cuenta de administrador. Con este usuario se podría activar o desactivar la extensión, añadir o quitar URLs de la lista negra, ver el feedback de los otros usuarios, etc.

Se podría implementar una notificación en tiempo real a los supervisores cuando un usuario este intentando consumir contenido peligroso. Así como un horario de funcionamiento configurable de la detección.

Además de todo esto, se podría ampliar el contenido que se filtra y entrenar modelos de otro tipo de contenido nocivo como pudieran ser las autolesiones, y que, de esta manera, el navegar por internet sea más seguro.

También se puede adaptar a otros navegadores para poder llegar al público que no utilice Google Chrome.

Y, por último, se puede realizar una app de Android e IOs para poder filtrar el contenido en dispositivos móviles que son los más utilizados por los jóvenes hoy en día.

6 BIBLIOGRAFÍA Y REFERENCIAS

[1] Martínez, Fátima. “Estudio Redes Sociales España IAB”. Accedido en Junio 2025, de https://fatimamartinez.es/2025/05/29/estudio-redes-sociales-espana-2025-iab/

[2] INE. “Población que usa Internet (en los últimos tres meses). Tipo de actividades realizadas por internet”. Accedido en Junio 2025, de https://www.ine.es/ss/Satellite?L=es_ES&c=INESeccion_C&cid=1259925528782&p=1254735110672&pagename=ProductosYServicios%2FPYSLayout

[3] Guan, Shuai. “80 Estadísticas Esenciales de TikTok para 2025”. Accedido en Junio 2025, de https://thunderbit.com/es/blog/tiktok-stats

[4] García López, Esther. “Facebook, Instagram y TikTok, las redes preferidas por los españoles, mientras X pierde un 36% de usuarios lastrada por Musk”. Accedido en Junio 2025, de https://www.infobae.com/espana/2025/06/26/facebook-instagram-y-tiktok-las-redes-preferidas-por-los-espanoles-mientras-x-pierde-un-36-de-usuarios-lastrada-por-musk/

[5] Newberry, Cristina. “30 Instagram statistics marketers need to know in 2025”. Accedido en Junio 2025, de https://blog.hootsuite.com/instagram-statistics/

[6] Yuste, María. “Instagram se ha propuesto luchar también contra la anorexia y la bulimia: a partir de ahora censurará todas las fotos que las inciten”. Accedido en Junio 2025, de https://www.trendencias.com/redes-sociales/instagram-se-ha-propuesto-luchar-tambien-anorexia-bulimia-a-partir-ahora-censurara-todas-fotos-que-inciten

[7] Safety Center TikTok. “Eating disorders”. Accedido en Junio 2025, de https://www.tiktok.com/safety/en/eating-disorder

[8] López. Alberto. “Control parental: todo lo que tienes que saber y cómo llevarlo a cabo”. Accedido en Agosto 2025, de https://www.redeszone.net/tutoriales/seguridad/control-parental-que-es-herramientas/

[9] Dilmegani, Cem. “Top 7 Next-Generation Firewall (NGFW) Features in 2025”. Accedido en Agosto 2025, de https://aimultiple.com/ngfw-features

[10] Internet Society. “Internet Society Perspectives on Internet Content Blocking: An Overview”. Accedido en Agosto 2025, de https://www.internetsociety.org/resources/doc/2017/internet-content-blocking/

[11] Zpedia. “¿Qué es un servidor proxi?”. Accedido en Agosto 2025, de https://www.zscaler.com/es/zpedia/what-is-a-proxy-server

[12] Rossi Lucia, Tizzano Paola, Malaspina Elisabetta, Moscano Filomena, Gualandi Paola, Rossi Francesca, Francia Valentina, Atti Anna-Rita, Parmeggiani Antonia. “Eating Disorders and the Internet: A Descriptive Cross-sectional Study Monitoring the Pro Ana Phenomenon in an Italian Sample”. Accedido en Agosto 2025, de https://www.longdom.org/open-access/eating-disorders-and-the-internet-a-descriptive-crosssectional-study-monitoring-thepro-ana-phenomenon-in-an-italian-samp.pdf

[13] Carmela Mento, Maria Catena Silvestri, Maria Rosaria Anna Muscatello, Amelia Rizzo, Laura Celebre, Martina Praticò, Rocco Antonio Zoccali, Antonio Bruno. “Psychological Impact of Pro-Anorexia and Pro-Eating Disorder Websites on Adolescent Females: A Systematic Review”. Accedido en Agosto 2025, de https://pmc.ncbi.nlm.nih.gov/articles/PMC7926357/

[14] Koncept. “Google Chrome lidera el mercado español”. Accedido en Junio 2025, de https://www.koncept.es/navegadores-mas-utilizados-espana/

[15] Chrome Developers. “Cómo migrar a Manifest V3”. Accedido en Agosto 2025, de https://developer.chrome.com/docs/extensions/develop/migrate?hl=es-419

[16] ITD Consulting. “La transición a Manifest V3 en Chrome: Implicaciones, cambios y controversias”. Accedido en Agosto 2025, de https://itdconsulting.com/sin-categoria/la-transicion-a-manifest-v3-en-chrome-implicaciones-cambios-y-controversias/

[17] Soares, Rebecca. “Actualizaciones de las políticas de Chrome Web Store: garantizamos claridad y coherencia para los desarrolladores”. Accedido en Agosto 2025, de https://developer.chrome.com/blog/cws-policy-updates-2025?hl=es-419

7 ANEXOS

Enlace al GitHub con todos los archivos del trabajo: https://github.com/kses99/TFG.git