Universitat Oberta de Catalunya

Generalización de consultas web para protección de privacidad de usuarios

Autor: Silvia Cobo Gómez
Director de TFG: Javier Parra Arnau

Grado de Ingeniería Informática
Ingeniería de Computadores

02/01/2020

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Generalización de consultas web para protección de privacidad de usuarios*
Nombre del autor:	*Silvia Cobo Gómez*
Nombre del colaborador/a docente:	*Javier Parra Arnau*
Nombre del PRA:	*Helena Rifà Tous*
Fecha de entrega:	*01/2020*
Titulación o programa:	*Grado de Ingeniería Informática*
Área del Trabajo Final:	*Seguridad Informática*
Idioma del trabajo:	*Español*
Palabras clave	*Seguridad, privacidad, web, plugin*

Resumen del Trabajo

El trabajo propone estudiar la implementación de un mecanismo para generalizar consultas web que opere en modo local, es decir, que se instale en el lado del usuario.
Para ello, se utilizará la herramienta *Wordnet* como base de conocimiento para la generalización de las consultas web, aunque de una forma reducida para no incurrir en problemas de espacio, almacenamiento o rendimiento en las búsquedas web.
Se evaluará, junto a una serie de usuarios voluntarios, el rendimiento de la aplicación, la privacidad que los usuarios perciben, el consumo de CPU memoria y el tiempo de procesamiento, para determinar la viabilidad y funcionalidad de esta.

Abstract:

In this project we propose to study the implementation of a tool that aims to generalize web search queries. This tool will operate locally, that is, on the user side.
For this, the *Wordnet* tool will be used as a knowledge base for the generalization of web queries, although in a reduced way so as not to incur space, storage or performance problems in web searches.
The performance of the application, the privacy perceived by users, the consumption of CPU memory and the processing time will be evaluated, together with a series of voluntary users, to determine its viability and functionality.

Índice

Figuras y tablas

Problema por resolver

En la actualidad, millones de usuarios utilizan a diario buscadores web para obtener información sobre cualquier asunto por el que sientan interés o sobre el que necesiten obtener más información. Para realizar estas búsquedas, los usuarios utilizan motores de búsqueda en sus navegadores de confianza. Motores tales como *Bing, Yahoo! Search,* etcétera. Estos motores de búsqueda pueden crear un perfil de usuario para ser capaces de, basándose en búsquedas anteriores del propio usuario, ofrecer los resultados más precisos al usuario en cuestión. Por ejemplo, si un usuario realiza a menudo consultas sobre planetas o el espacio y busca la palabra Mercurio, los resultados referentes al planeta aparecerán en primer lugar, sobre aquellos que se refieran al elemento químico. Este perfil permite que la experiencia de usuario sea más satisfactoria, dado que permite que las búsquedas ofrezcan el resultado esperado con más rapidez.

No obstante, la creación y utilización de estos perfiles de usuario plantea un problema de privacidad. Nuestro perfil de usuario, en definitiva, nuestros datos, están disponibles para el buscador en Internet, en su base de datos. ¿Qué pasaría si algún usuario malintencionado pudiera acceder a estos datos y, de alguna manera, acabar adivinando nuestra identidad real? ¿Existe alguna forma de proteger nuestra privacidad y, a la vez, seguir obteniendo resultados satisfactorios en las búsquedas?

En el presente trabajo se propone estudiar la implementación de un mecanismo para generalizar consultas web que opere en modo local, es decir, que se instale en el lado del usuario. La aplicación realizará una generalización sobre el término que el usuario desea buscar antes de que el usuario ejecute la búsqueda, y la sustituirá por el término generalizado en el buscador. De este modo, se pretende proteger la privacidad del usuario con respecto a la búsqueda deseada, al haber sido modificada previamente. La generalización de la palabra permitirá obtener resultados semánticamente similares, pero con una capa de protección de la privacidad extra para el usuario. En concreto, en este trabajo se permitirá proteger la privacidad en el buscador web de Google en el navegador Google Chrome.

Se utilizará la herramienta *Wordnet* como base de conocimiento, para permitir la generalización de las consultas web del usuario.

Se evaluará, junto a una serie de usuarios voluntarios, el rendimiento de la aplicación, la privacidad que los usuarios perciben, el consumo de CPU memoria y el tiempo de procesamiento, para determinar la viabilidad y funcionalidad de esta.

Estado del arte

En la literatura, observamos que ya existen algunos métodos de generalización y falsificación de consultas [1] que permiten reforzar la privacidad de los perfiles de usuario en búsquedas web. Por ejemplo, existen métodos para falsificar las consultas en base a la información obtenida en el perfil de alguna red social del mismo usuario, por ejemplo, Twitter. Utilizando los datos de Twitter se crea una base de conocimiento del propio usuario que permite falsificar las consultas web con datos lo más similares posibles y adaptados al perfil del usuario, pero que no permitirían que un atacante que accede a ese perfil de usuario pudiera encontrar su identidad real en base a los datos obtenidos del perfil.

A pesar de existir algunas ideas y herramientas que explotan diferentes posibilidades de privacidad [2][3], ninguna de ellas se ha planteado hasta el momento el instalar en el lado del usuario esta herramienta, en forma de plugin en el navegador web. Por lo tanto, esto es lo que vamos a estudiar en este trabajo.

Objetivos generales

El proyecto consiste en estudiar e implementar una forma de generalizar consultas web para proteger la privacidad del usuario que realiza estas búsquedas y, posteriormente, evaluar la viabilidad de la herramienta desarrollada.

En concreto, esta forma de generalizar las consultas se llevará a cabo utilizando un plugin para el navegador Chrome, instalado en la parte del cliente, es decir, en la máquina del usuario que realiza las consultas.

Los objetivos generales, son:

- Programar un *plugin* o aplicación que permita administrar el nivel de protección de la privacidad del usuario en cuanto a consultas web, aportando diferentes niveles de privacidad a su perfil.

- No debe ser necesario ampliar el hardware de la máquina del cliente, por ello todo el peso de la operativa debe realizarse en la nube o en infraestructura externa, propia de la aplicación.

- El *plugin* debe instalarse en el lado del cliente, es decir, en su propio navegador.

- La utilización *plugin* no debería afectar al rendimiento de la máquina del cliente ni aumentar excesivamente el tiempo de búsqueda de los términos en el buscador.

- Evaluar la viabilidad de esta aplicación utilizando una batería de pruebas con la colaboración de varios usuarios de prueba, que nos permitirán detectar los puntos fuertes y débiles de la misma.

- Recoger los comentarios de usuarios y los propios para elaborar unas conclusiones y ofrecer una idea de versión mejorada teniendo en cuenta todos los puntos anteriores.

Metodología y proceso de trabajo

La herramienta que vamos a utilizar como base para nuestro proyecto es *WordNet*. Se trata de una base de datos que contiene palabras, sus definiciones y sus relaciones semánticas. Al ser un producto final ya disponible para su descarga y utilización, partimos de una base sólida para comenzar todo el trabajo de desarrollo a su alrededor.

Wordnet nos permitirá generalizar las búsquedas realizadas en la web, para adecuar el nivel de privacidad a elección del usuario. Por ejemplo, nos permitiría generalizar la búsqueda de alguna enfermedad o trastorno, como podría ser el TOC (trastorno obsesivo-compulsivo) a “enfermedad mental” o, incluso, a enfermedad simplemente, dependiendo del nivel de privacidad indicado.

Tras haber analizado *Wordnet*, decidimos establecer dos niveles de privacidad para el usuario: uno más simple, para enmascarar ligeramente la consulta, y otro que ofrezca un nivel máximo de privacidad (a pesar de salir perdiendo en cuanto a lo acertado de los resultados).

Se realizará una función en lenguaje *Python* para consultar la base de datos de *Wordnet*. El programa recogerá los términos de la búsqueda realizará por el usuario y, aplicando los niveles de privacidad requeridos, cambiará los mismos para regenerar la consulta que se va a enviar a Google.

La prueba de concepto del programa consistirá en programar un *plugin* para el navegador web Google Chrome. Una vez elaborada la primera versión de este, se concertarán entrevistas con varios usuarios voluntarios que nos permitirán instalar el software en sus máquinas y realizarán una serie de pruebas preestablecidas para evaluar el funcionamiento y rendimiento de la herramienta. Una vez realizadas las pruebas preestablecidas, se les dejará también realizar pruebas a libre albedrío.

Una vez tengamos las conclusiones por parte de los usuarios, pasaremos a finalizar la memoria con las conclusiones del proyecto, recogiendo los puntos fuertes, los que se podrían mejorar, lecciones aprendidas, etc.

Planificación

Como puede verse en el diagrama, se han desgranado y planificado las tareas de la siguiente forma:

TAREA	INICIO	FIN	DÍAS	ESTADO
Planificación proyecto (PEC 1)	9/18	10/7	19	Completo
Análisis Wordnet (PEC 2)	10/1	11/4	34	En desarrollo
Desarrollar plugin (PEC 3)	11/4	12/2	28	No comenzado
Pruebas plugin - usuarios	12/2	12/15	13	No comenzado
Elaborar memoria (PEC 4)	12/2	1/2	31	No comenzado
Elaborar presentación virtual	1/3	1/9	6	No comenzado

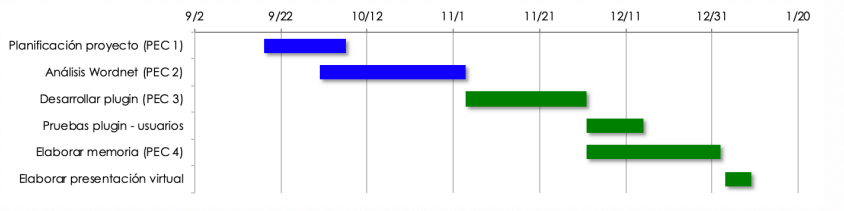


Figure 1: Planificación

- Planificación Proyecto, que se corresponde con la primera entrega (PEC 1), comprende la elaboración y entrega del presente informe, donde se detallan los puntos solicitados en el enunciado:
    - La explicación detallada del problema a resolver.
    - La enumeración de los objetivos que se pretenden alcanzar con la realización del TFG.
    - La descripción de la metodología que se seguirá durante el desarrollo del TFG.
    - El listado de las tareas a realizar para alcanzar los objetivos descritos.
    - La planificación temporal detallada de estas tareas y sus dependencias.
    - Una pequeña revisión del estado del arte.

- Análisis Wordnet, que se corresponde con la PEC 2. Para poder utilizar y sacar el máximo rendimiento de la herramienta, se dedican varias semanas de la planificación en exclusiva a estudiar y entender la herramienta y sus funcionalidades. Se pretende:
    - Comprender el uso de la herramienta para obtener las generalizaciones de la forma más correcta posible.
    - Reducir el tamaño de la base de datos con los datos necesarios para que las consultas sean más ligeras.
    - Realizar una guía con lo que se ha aprendido y las modificaciones realizadas a nivel de base de datos, para aprovechar esta información posteriormente en la memoria.

- Desarrollar plugin, que se corresponde con la PEC3. A pesar de que se irá avanzando en este tema mientras se analice *Wordnet*, a partir de esta fecha los esfuerzos irán orientados a la programación y optimización del plugin de Chrome. Los pasos por seguir serán:
    - Elaboración del código en *Python* para transformar consultas introducidas directamente en el programa.
    - Se introducirá directamente en el código el nivel de privacidad requerido y se realizarán pruebas de viabilidad.
    - Cuando esto funcione de manera óptima, pasaremos a recoger las consultas directamente de la web
    - Devolveremos la transformada de la consulta para ser enviada a Google

  - Se realizará una pequeña interfaz gráfica para que el usuario pueda modificar el nivel de privacidad. En un principio se establece privacidad básica y máxima, pendiente de evaluar tras el estudio de *Wordnet* si se podrán añadir más niveles.
  - Se realizarán las pruebas necesarias para certificar el funcionamiento
  - Se elaborará un manual de desarrollo y utilización del plugin, que se adjuntará en la memoria.

- Pruebas del plugin por parte de usuarios. Esta sección se llevará en paralelo con la siguiente, puesto que no precisa de mi intervención activa. Contaremos con 2-3 usuarios para que testen y evalúen el plugin
  - Se elaborará un listado con los usuarios y su consentimiento para participar en el testeo
  - Se les ayudará a instalar el plugin y se les proporcionará un listado de tareas mínimas para evaluar
  - Se recogerán las conclusiones unos días más tarde
  - Se elaborará un informe para ser incluido en la memoria. Asimismo, si se encuentra algún fallo o posible mejora que se pueda aplicar, se estudiará su desarrollo e implementación en el tiempo restante.

- Elaboración de la memoria, correspondiente con la PEC4. En este punto, y a pesar de que ya hemos ido elaborando documentación en los apartados anteriores, nos centraremos en la elaboración de la memoria del proyecto y en explicar todos los puntos de forma correcta, adecuada y concisa.
  - Revisión de los puntos ya documentados para aplicar correcciones
  - Documentación de los puntos pendientes correspondientes a pruebas y conclusiones
  - Revisión de la correcta inclusión y etiquetado de tablas, figuras, citas, etc.
  - Revisión general del documento para confirmar que cumple todos los puntos necesarios.
  - En este punto entregaremos también el plugin, por lo que se adjuntará en la entrega la versión final del mismo

- Elaboración de la presentación virtual: Una vez entregado el producto y la memoria, trabajaremos en la elaboración de la presentación virtual. Esta incluirá:
  - Una presentación *powerpoint* con un resumen del proyecto: introducción, objetivos, metodología de trabajo, resultados de pruebas realizadas con usuarios, conclusiones, etc.
  - Una demostración del funcionamiento del producto

Wordnet

Wordnet es un proyecto creado en 1985 [4] y que sigue siendo mantenido desde entonces por la Universidad de Princeton, en concreto por el '*Cognitive Science Laboratory'*. Consiste en una base de datos léxica del idioma inglés, cuya descarga se ofrece de forma abierta y gratuita a los usuarios que deseen utilizarlo.

El propósito de *Wordnet* consiste en la agrupación de sustantivos, adjetivos, etc. en conjuntos de sinónimos cognitivos, que ellos denominan *synsets*. Los *synsets* se conectan entre ellos por medio de relaciones conceptuales-semánticas y léxicas, dando como resultado una red de palabras y conceptos con aplicaciones diversas.

Entre estas aplicaciones se encuentran clasificaciones automáticas de diferentes textos, traducciones automáticas, generación de crucigramas, resúmenes de texto, etc.

En nuestro caso, utilizaremos la base de datos de generalización de las palabras, también conocido como hiperónimos. Buscaremos el *synset* de la palabra que se desea buscar y obtendremos el hiperónimo más adecuado siguiendo ciertas reglas que se han estudiado previamente.

Función para generalizar consultas

Para esta prueba de concepto, nos hemos centrado en búsquedas de una sola palabra, es decir, no se realizan búsquedas de frases. A pesar de que este proceso se puede paralelizar para permitir generalizar la búsqueda de una frase, la generalización de palabras separadas puede alejarse demasiado del término inicial que se deseaba buscar. Además, el tiempo incrementaría a la hora de montar la respuesta para poder enviar la búsqueda a Google.

Utilizaremos los hiperónimos de *Wordnet* para generalizar las consultas realizadas por los usuarios, permitiendo generalizar en un nivel básico y en un nivel más restrictivo.

Se han tomado varias decisiones para agilizar y simplificar el proceso.

1- Simplificación de consultas: Cuando se realiza una consulta a *Wordnet*, aparecen resultados para todos los posibles significados de esa palabra. Como no tenemos ninguna información sobre el perfil de usuario en nuestra aplicación (de hecho, es lo que queremos evitar tener), no podemos saber exactamente a cuál de los significados se refiere. Por lo tanto, asumiremos que el primero de todos los resultados devueltos es el correcto.

Si realizamos una consulta buscando *aids* que serían las siglas en inglés para la enfermedad del SIDA, el primer resultado se refiere a la enfermedad, que es lo que estamos buscando. El resto de los resultados se refieren a otros significados tales como trabajo, persona, soporte… que podrían ser el objetivo de búsqueda de algún usuario pero que por regla general no será así.

2- Agilizar la generalización: Una vez elegido el significado, podemos obtener un listado de hiperónimos, es decir, de generalizaciones para el mismo. A veces, esta búsqueda devuelve un único significado y otras devuelve un listado.

Tras un estudio previo, se ha decidido que la primera palabra de la lista se corresponde con un nivel bajo de generalización y, así como leemos los resultados que va ofreciendo, la generalización es cada vez mayor.

Por ejemplo, siguiendo con el ejemplo del SIDA, el primer nivel de generalización devolvería una consulta a “inmunodeficiencia” y el nivel más restrictivo “enfermedad infecciosa”.

Por el momento no se ha modificado la base de datos de *Wordnet* para reducirla y así reducir también el tiempo de respuesta, puesto que la complejidad de la ejecución de la tarea no compensa la ganancia de tiempo de consulta y apenas añade unas décimas de segundo a la búsqueda.

El código implementado es bastante sencillo, y únicamente ejecuta los puntos comentados anteriormente: la función recibe la palabra a generalizar y el nivel de generalización deseado.

```
def generalize_search(word, basicPrivacy):
    if wn.synsets(word):
        results = [lemma.name() for synset in wn.synsets(word)[0].hypernyms() for lemma in synset.lemmas()]
        if basicPrivacy:
            return results[0].replace("_", " ")
        else:
            return results[len(results) - 1].replace("_", " ")
    return word
```

Con estos datos, busca la generalización y devuelve la palabra generalizada adecuada. En el caso de no existir la palabra en la base de datos de *Wordnet*, devuelve la misma palabra a buscar sin modificar, para que el usuario pueda realizar la búsqueda sin perder más tiempo.

En resumen, la función realiza los pasos contenidos en el siguiente diagrama:

- Recoge como input la palabra
- Realiza la consulta a *Wordnet*
- Recoge el primer significante de la palabra
- Obtiene la primera y última generalización de la palabra en cuestión
- Consulta si el usuario deseaba la generalización básica o estricta
- Devuelve la generalización solicitada, si está disponible

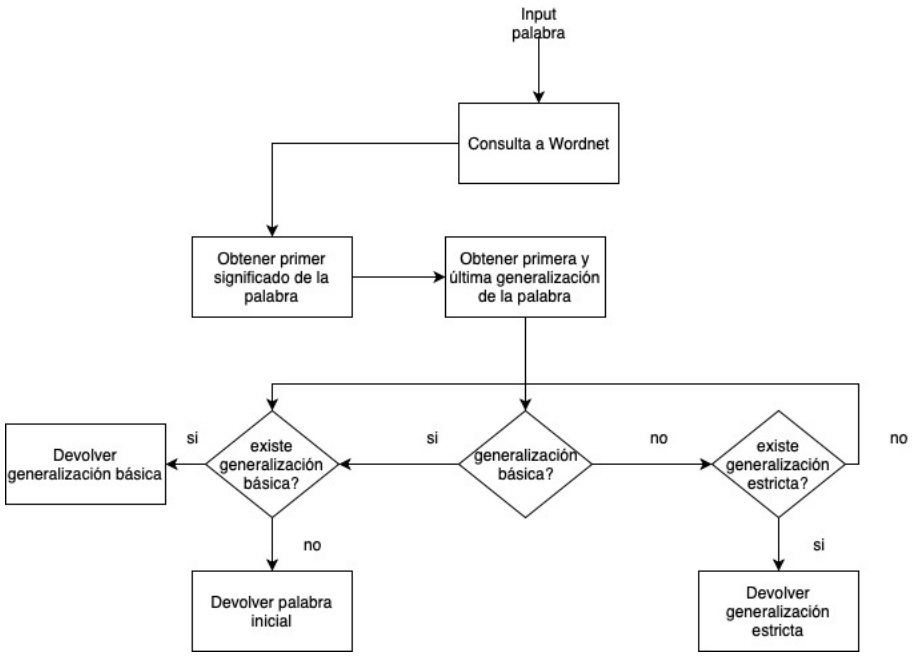


Figure 2: Diagrama flujo función python

Extensión Google Chrome

Como habíamos comentado anteriormente, debemos proporcionar algún tipo de herramienta para que el usuario final pueda realizar sus pruebas en la herramienta desarrollada. En este caso, hemos decidido realizar una extensión instalable en el navegador Google Chrome.

Estas extensiones permiten ejecutar funciones y programas predeterminados en el navegador Google. También se puede elegir el momento de ejecución, por ejemplo, al abrir una nueva pestaña, al cerrarla, al ejecutar una búsqueda…

Las extensiones de Google Chrome consisten en una carpeta en la que se encuentran diferentes archivos. Uno de ellos es el archivo *manifest.json*, donde se incluye la información básica de la extensión o *plugin* y los permisos necesarios, el momento en el que se tiene que ejecutar el *plugin*, etc.

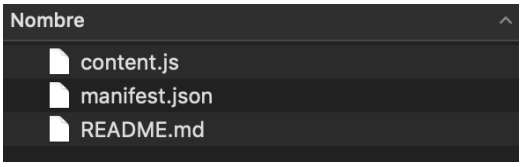


Figure 3: Estructura de carpetas plugin

El fichero más importante es el fichero de contenido (*content.js*). Consiste en un código *javascript* que contiene los pasos a ejecutar para ejecutar la funcionalidad deseada. En nuestro caso, capturar el texto del buscador, consultar la función que, a su vez, consulta *Wordnet*, y devolver el resultado a google justo antes de ejecutar la búsqueda, para que así se sustituya por el término generalizado.

Actualmente el código *javascript* recoge cualquier cambio que se realiza en el cuadro de texto del buscador de google, envía esta palabra a un servidor que se conecta a la función *Python* que comentábamos en el apartado anterior, y ésta devuelve la generalización de la palabra, según el nivel de privacidad deseado.

Para consultar la función *Python* encargada de devolver la generalización, utilizamos un servidor web ubicado en la nube, en concreto en *aws*. Este servidor alberga el código *Python* y un servidor web que escucha en el puerto 8787 las consultas, y ofrece el resultado de estas.

La infraestructura del servidor *aws* consta internamente de:

- Un servidor https para servir las consultas al código *Python*. Para ello utiliza un *nginx* como primer punto de contacto y receptor de las consultas. Es en este punto donde hemos creado un certificado auto-firmado para poder cumplir los estándares de Google y ofrecer protección *ssl* en las consultas.
- Un servidor web https que recoge la información recibida vía *nginx* mediante un puerto distinto y que, a su vez, consulta la función *Python*

- La función *Python* en sí, que ya se ha explicado en apartados anteriores, que recoge palabra y nivel de privacidad, y devuelve la información correspondiente.

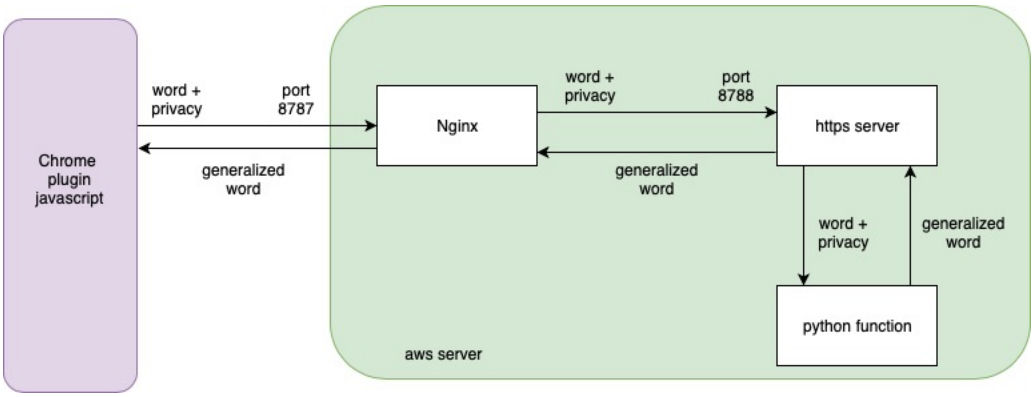


Figure 4: Infraestructura servidor aws

El nivel de privacidad de la consulta se establece en el propio *plugin*, concretamente en el fichero *content.js*, en la línea que contiene la URL del servidor web. Si escribimos *True*, tenemos privacidad básica y, en caso de desear una privacidad más estricta, podemos indicar *False:*

```
xhttp.open("GET", "https://35.180.253.208:8787/"+this.value+"/True", false);
```

El código completo del fichero *content.js* es:

```
var original_search = document.getElementsByClassName('gLFyf gsfi')[0];

if (original_search){
console.log(original_search);
original_search = original_search.addEventListener('change',function(evt){
    var xhttp = new XMLHttpRequest();
    xhttp.open("GET", "https://35.180.253.208:8787/"+this.value+"/True", false);
    xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      document.getElementsByClassName('gLFyf gsfi')[0].value = this.responseText;
    }
  };
  xhttp.send();
});
}
```

La función *Python* a la que se llama, recibe la palabra y el nivel de privacidad y devuelve la generalización solicitada en base a esos parámetros.

Se contempla el caso de que sólo devuelva un resultado de generalización y también se contempla el caso en el que la palabra no exista en la base de datos. En ese caso, se devuelve como resultado la palabra original.

El nivel de privacidad se debe establecer en el código del *plugin*, y se puede modificar directamente en el ordenador de cada uno de los participantes en el estudio. Se les permite cambiar el nivel de privacidad accediendo al código del *plugin* y recargándolo ellos mismos, puesto que es una tarea sencilla. No obstante, se ha considerado fuera del alcance las pruebas con usuarios y se ha ejecutado por mi tras una semana de pruebas en cada modo.

El producto final

El resultado final de esta prueba de concepto es una extensión de google Chrome que utiliza un algoritmo basado en *Wornet* para permitir la generalización de la consulta en base a los parámetros dados y la palabra deseada. El funcionamiento a nivel lógico del aplicativo puede verse en el siguiente diagrama:

- El usuario introduce su búsqueda en el buscador del navegador web Google Chrome
- El *plugin* entra en acción y ejecuta su secuencia
- Llama al servidor web para obtener la generalización solicitada
- El servidor web provee el listado de resultados
- Se recorre este listado para descartar los resultados no deseados: si se desea aplicar privacidad básica, se descartan los resultados de privacidad estricta, etc.
- Si la generalización deseada está disponible, el *plugin* devuelve esta palabra y ésta se busca en Google
- En el caso de que la generalización deseada no esté disponible, se ofrecerá la siguiente alternativa disponible. Es decir, si se desea privacidad estricta pero solo se encuentra privacidad básica, se devolverá esta última. Si tampoco se obtiene generalización básica, se devuelve la búsqueda inicial.

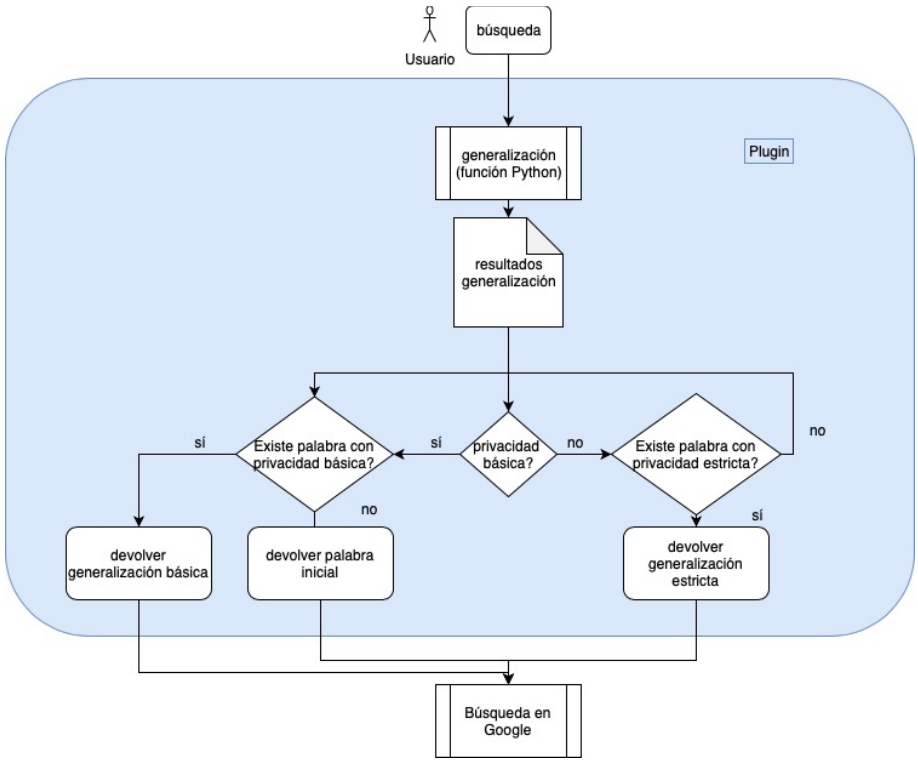


Figure 5: Diagrama flujo plugin

Por lo tanto, en el proceso de búsqueda, la extensión se encarga de realizar una modificación en el momento en que el usuario pulsa el botón de búsqueda y cambia la palabra a buscar por la generalización de la misma, en el caso de que esta exista.

Manual instalación *plugin*

Para instalar el *plugin* únicamente se debe descomprimir la carpeta e introducir la siguiente ruta en el navegador Google Chrome:

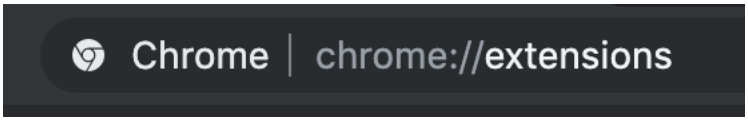


Figure 6: Instrucciones instalación 1

Posteriormente, aparece el menú de extensiones o *plugins* del navegador. En la parte superior aparecen diversas opciones, elegiremos la opción correspondiente a “Cargar descomprimida”, para cargar la carpeta que acabamos de descomprimir.

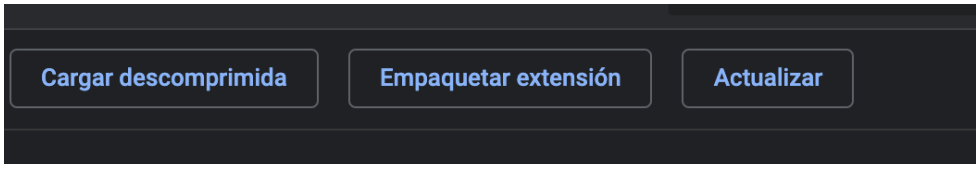


Figure 7: Instrucciones instalación 2

Buscamos en nuestro ordenador la carpeta correspondiente y la cargamos. Una vez hecho este paso, en el menú de extensiones disponibles, aparecerá nuestro *plugin* de generalización de búsquedas, como en la siguiente imagen:

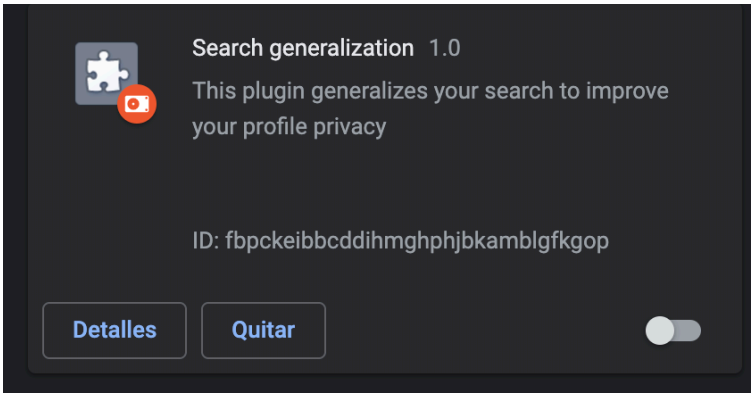


Figure 8: Instrucciones instalación 3

Los *plugin* de google Chrome no permiten que se utilicen llamadas http puesto que no se consideran seguras. Por lo tanto, en el servidor web que ofrece las generalizaciones de *Wordnet*, hemos creado unos certificados auto

firmados para poder utilizar https. Al no ser certificados de una entidad profesional, antes de empezar a utilizar el *plugin*, tenemos que aceptar el certificado. Por lo tanto, es importante introducir la siguiente URL y confirmar en el navegador que se confía en este certificado

https://35.180.253.208:8787

Tras este gesto, ya podemos empezar a utilizar el *plugin* en nuestro navegador. Cabe destacar que únicamente funcionará si buscamos en el cuadro de búsqueda, en la pantalla representada en la siguiente imagen. Si introducimos la búsqueda directamente en el navegador, no estaremos utilizando el *plugin*.

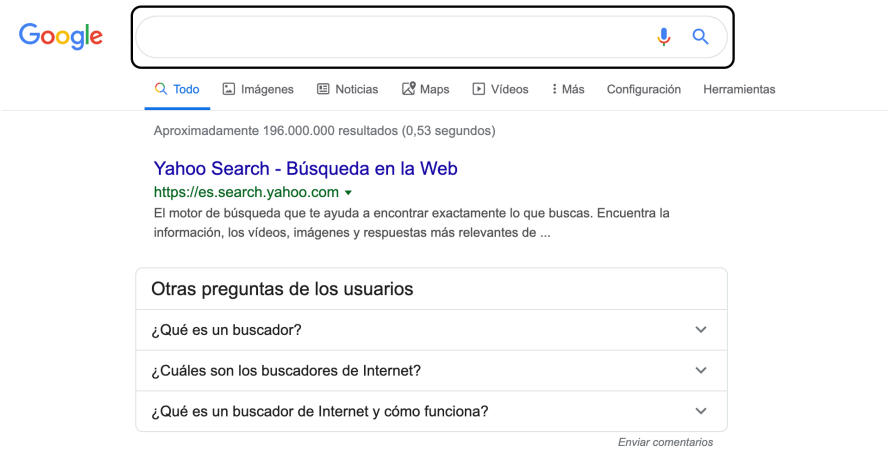
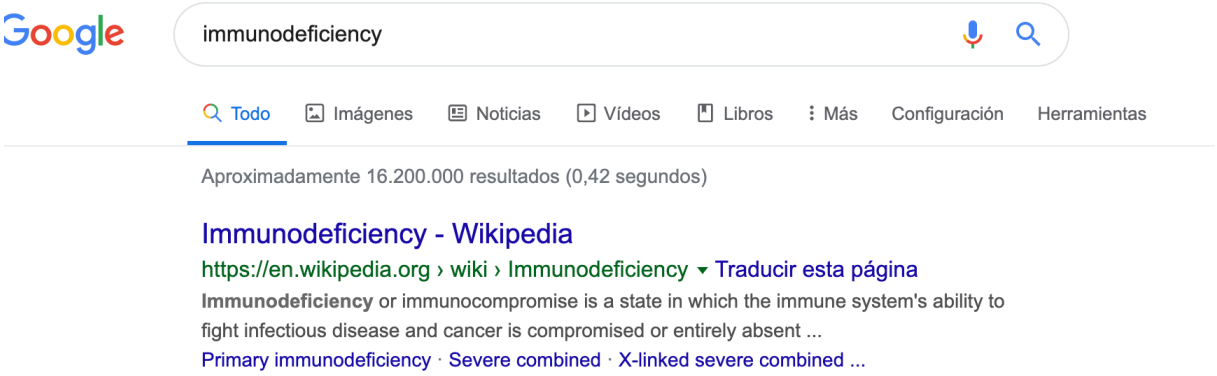


Figure 9: Instrucciones instalación 4

Cuando introducimos la palabra a buscar y pulsamos la tecla Intro, automáticamente vemos como cambia el texto a buscar según la privacidad seleccionada



Pruebas con usuarios

Hemos contado con la colaboración de los siguientes perfiles y usuarios:

Perfiles técnicos:

- Oscar Cobo Gómez, programador Java
- José Laredo López, ingeniero en informática de sistemas
- Daniel Nieto, técnico informático

Perfiles no técnicos:

- Ana María Gómez Lázaro, camarera de piso
- Delia Martínez Flores, quiromasajista

Los ordenadores utilizados para las pruebas son ordenadores de mesa básicos o portátiles estilo *netbook*, es decir, que no poseen grandes prestaciones técnicas. El uso de este tipo de máquinas no debería penalizar en ningún caso el uso del *plugin*, puesto que este se centra en el uso web y en el servidor que realiza las consultas a *Wordnet* que se encuentra en la nube.

Ambos grupos de usuarios han realizado una batería de búsquedas con el *plugin* en modo privacidad básica y privacidad estricta. Las pruebas consistían en realizar una serie de consultas sobre diversas enfermedades en google y evaluar la rapidez en la ejecución de estas (1-5) y la satisfacción con los resultados obtenidos (1-5). Seguidamente, también se les pide que utilicen el *plugin* libremente para obtener sus impresiones

Privacidad básica

Table 1: Resultados privacidad básica

Búsqueda	Perfil técnico						Perfil no técnico			
Privacidad básica	Perfil 1		Perfil 2		Perfil 3		Perfil 1		Perfil 2	
	Resultado	Rapidez	Resultado	Rapidez	Resultado	Rapidez	Resultado	Rapidez	Resultado	Rapidez
Herpes	4	5	4	5	4	5	2	5	3	5
Lupus	5	5	5	5	5	5	4	5	4	5
Epilepsy	5	5	5	5	5	5	3	5	3	5
Flu	5	5	5	5	4	5	4	5	4	5
Ebola	4	5	3	5	4	5	3	5	3	5
Agoraphobia	4	5	4	5	4	5	4	5	2	5
Bulimia	4	5	3	5	5	5	2	5	3	5
Depression	4	5	3	5	4	5	3	5	2	5

Cancer	5	5	5	5	5	5	5	5	5	5
Ictus	1	5	2	5	2	5	1	5	1	5
Nota media	4’1/5	5/5	3’9/5	5/5	4’2/5	5/5	3’1/5	5/5	3/5	5/5

Los usuarios otorgan una media de 5/5 en rapidez en las búsquedas y una media de 3’66/5 en la precisión de las búsquedas

La media de precisión los usuarios técnicos es de 4’1/5 y de los usuarios no técnicos es de 3’05/5. Por lo tanto, los usuarios técnicos han obtenido resultados más satisfactorios que los usuarios no técnicos.

Privacidad estricta

Table 2: Resultados privacidad estricta

Búsqueda	Perfil técnico						Perfil no técnico			
Privacidad estricta	Perfil 1		Perfil 2		Perfil 3		Perfil 1		Perfil 2	
	Resultado	Rapidez	Resultado	Rapidez	Resultado	Rapidez	Resultado	Rapidez	Resultado	Rapidez
Herpes	4	5	4	5	4	5	2	5	3	5
Lupus	4	5	5	5	5	5	3	5	3	5
Epilepsy	4	5	5	5	5	5	3	5	3	5
Flu	4	5	5	5	3	5	3	5	3	5
Ebola	2	5	2	5	2	5	1	5	1	5
Agoraphobia	4	5	4	5	4	5	2	5	3	5
Bulimia	4	5	3	5	5	5	2	5	3	5
Depression	4	5	3	5	4	5	3	5	3	5
Cancer	5	5	5	5	5	5	5	5	5	5
Ictus	1	5	2	5	1	5	1	5	1	5
Nota media	3’6/5	5/5	3’8/5	5/5	3’8/5	5/5	2.5/5	5/5	2’8/5	5/5

Los usuarios otorgan una media de 5/5 en rapidez en las búsquedas y una media de 3’3/5 en la precisión de las búsquedas

La media de precisión los usuarios técnicos es de 3’73/5 y de los usuarios no técnicos es de 2’65/5. Por lo tanto, los usuarios técnicos han obtenido resultados más satisfactorios que los usuarios no técnicos, al igual que en el caso anterior, con las búsquedas en modo privacidad básica.

Comparativa satisfacción según privacidades

Tras revisar los resultados del apartado anterior y ver que todos los usuarios nos han dado la nota máxima en rapidez de ejecución de las consultas, vamos a pasar a analizar únicamente las satisfacciones con respecto al resultado obtenido.

En este caso, podemos ver que hay discrepancias entre las opiniones de los diferentes usuarios, no obstante, sus resultados son concordantes con el resto de los usuarios de su mismo perfil así que procedemos a agruparlos en el siguiente gráfico:

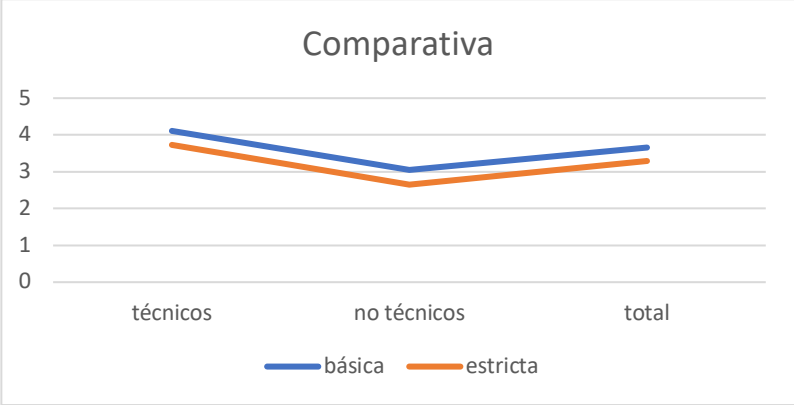


Figure 10: Comparativa usuarios

En ambos casos, los usuarios técnicos ofrecen puntuaciones más altas, tanto para privacidad básica como para privacidad estricta, y los usuarios de perfil no técnico puntúan más bajo. No obstante, a pesar de las puntuaciones más bajas, el resultado medio sigue siendo una nota de aprobado (superior a 2'5 sobre los 5 puntos totales asignables)

Conclusiones test con usuarios

Según las puntuaciones asignadas por los usuarios, obtenemos las siguientes conclusiones:

- Los usuarios de ambos perfiles consideran que los resultados aparecen con bastante rapidez, es decir, que no se añade tiempo de espera excesivo al utilizar el *plugin*. Indican que no encuentran apenas diferencia entre una búsqueda normal y una búsqueda con el *plugin*. Por lo tanto, parece que cumplimos con el objetivo de no añadir tiempo ni consumir recursos de la máquina del cliente.
- Los resultados son más satisfactorios con el filtro de privacidad básico, como era de esperar. No obstante, en algunas pruebas, algunos usuarios consideran que la búsqueda con privacidad estricta cumple mejor

sus expectativas con respecto a la búsqueda con privacidad básica. En este caso, esto también podría ser predecible, puesto que no todos los usuarios tienen el mismo tipo de expectativas con respecto a una palabra que el resto, y la generalización estricta puede acercarse más al concepto que estos tienen de la palabra.

- En general, los usuarios con perfil técnico están más satisfechos con los resultados que los usuarios con perfil no técnico. Esto puede deberse a que, al estar familiarizados con lo que se ha hecho en el *plugin*, la tecnología utilizada y sus restricciones, sean más comprensivos con el proceso que sigue el programa.

Los comentarios recogidos en cuanto a las pruebas libres se refieren a que es complicado hacer vida normal con este *plugin*, dado que los resultados siempre salen algo desvirtuados y les dificulta encontrar lo que buscaban en un inicio, haciéndoles invertir más tiempo del necesario e incluso teniendo que cambiar de navegador a otro sin el *plugin* para obtener los resultados que desean, porque no quieren buscar las palabras en inglés. Además, el problema de no poder buscar frases completas resta más puntos a la experiencia de usuario.

Los perfiles no técnicos concluyen en que no lo utilizarían en su día a día. Les resulta poco útil y no están acostumbrados a tener que cambiar el idioma a inglés para sus búsquedas. Como ya hemos mencionado anteriormente, el problema de no poder buscar frases también les dificulta el uso habitual en su día a día.

Los usuarios de perfil técnico coinciden en que lo utilizarían en su ordenador privado cuando se incluyera la posibilidad de buscar frases pero que, para sus búsquedas profesionales, referidas a términos técnicos de programación/sistemas, no ofrece un uso satisfactorio. Les ha gustado mucho como funciona el *plugin* y estarían dispuestos a seguir participando en pruebas si se quisiera continuar el desarrollo.

Conclusión

Tras realizar la programación de la prueba de concepto del *plugin* y las pruebas con usuarios, podemos concluir que la idea estudiada en este trabajo constituye una buena base para seguir trabajando en el futuro.

Los resultados son buenos: el tiempo de consulta para la generalización es prácticamente despreciable y, en general, la satisfacción de los usuarios con los resultados obtenidos es bueno. No obstante, se recogen una serie de puntos de mejora de cara a una segunda versión del *plugin*.

Los primeros puntos de mejora detectados se centran en la herramienta *Wordnet* en sí. La primera limitación detectada es la barrera del idioma, puesto que actualmente sólo se permite realizar consultas en inglés. Si bien se han encontrado algunas referencias a adaptaciones de la base de datos de *Wordnet* al español [5], en el momento no existe ninguna alternativa de código libre disponible. Por lo tanto, centrarse en encontrar o elaborar una adaptación de la base de datos de *Wordnet* al español podría ser un primer paso para mejorar el trabajo realizado.

Por otra parte, siguiendo con *Wordnet*, la siguiente limitación se refiere a la capacidad de poder consultar frases completas o varias palabras. En este momento, únicamente se pueden buscar palabras por separado. Se debería modificar el *plugin* para permitir generalizar más palabras en la misma consulta, a pesar de que esto suponga realizar varias consultas a *Wordnet*. Un problema derivado de esto es la posible desvirtuación de la búsqueda, al generalizar cada una de las palabras por separado, la probabilidad de que el resultado no tenga sentido una vez generalizado es muy alta. Por lo tanto, es un apartado a estudiar en otra prueba de concepto, para evaluar la mejor forma de salvaguardar este obstáculo.

En tercer lugar, también relacionado con *Wordnet*, tenemos un problema relacionado con el listado de hiperónimos, es decir, de las generalizaciones. Hemos asumido que la primera palabra que devuelve el listado es la generalización básica y que la última sería la de mayor generalización, pero no ocurre así en todos los casos, puesto que asumimos un significado de la palabra que puede no ser el que quiera el usuario. Se debería rehacer un estudio de este listado y, además, tal vez ofrecer una generalización de más niveles.

Relacionado también con el punto anterior, hay muchas palabras que únicamente devuelven una palabra en el listado de generalizaciones. Por lo tanto, otro punto de mejora a añadir sería el de revisar la base de datos para poder ampliarla con más información. Al ser una tarea que debe hacerse a mano directamente en la base de datos descargada en local, podría ser una tarea muy tediosa. Como hemos comentado anteriormente, se debería revisar la base de datos de *Wordnet* para adaptarla al idioma español y, de paso, podríamos preparar un acceso mediante api o similar, para poder modificar la base de datos de palabras mediante alguna web o llamada a este api, para así poder añadir o borrar información. Si exportamos la base de datos a *mysql*, podríamos acceder por un simple portal web para añadir las informaciones que fueran necesarias de una forma más cómoda.

Para acabar, en este momento tenemos una función *Python* que se ejecuta en un servidor web en la nube, concretamente en la nube de Amazon (AWS). La máquina no precisa de una gran infraestructura, por lo tanto, de

momento su uso es gratuito, es decir, no nos incurre en ningún tipo de coste. Pero, si queremos modificar la base de datos, deberíamos buscar donde alojarla y, en este caso, si será necesario invertir algo de presupuesto en la infraestructura necesaria.

El estudio y la realización de estas mejoras, y su posterior evaluación con usuarios, supondría un avance tal que la aplicación o *plugin* podría ofrecerse al mercado de *plugins*, para así ser utilizado por usuarios finales alrededor del mundo, y que este se introdujera en el día a día de muchos usuarios que deseen proteger su privacidad.

Bibliografía

[1] Javier Parra-Arnau, David Rebollo-Monedero, Jordi Forné, "Optimal Forgery and Suppression of Ratings for Privacy Enhancement in Recommendation Systems," (MDPI) Entropy, vol. 16, no. 3, Mar. 2014, pp. 1586-1634. DOI: 10.3390/e16031586.

[2] Alexandre Viejo, David Sánchez, "Profiling Social Networks to Provide Useful and Privacy-Preserving Web Search", Journal of the Association for Information Science and Technology, vol. 65, no. 12, pp. 2444-2458, 2014.

[3] David Sánchez, Jordi Castellà-Roca, Alexandre Viejo, "Knowledge-Based Scheme to Create Privacy-Preserving but Semantically-Related Queries for Web Search Engines", Information Sciences, vol. 218, pp. 17-30, 2013.

[4] Princeton University, “Wordnet, A Lexical Database for English”, Recuperado de: https://wordnet.princeton.edu

[5] Ana Fernández, Glòria Vázquez, Irene Castellón. (2008). Spanish WordNet 3.0. Spain. Red Temática en Tratamiento de la Información Multilingüe y Multimodal (TIMM). Recuperado de: http://timm.ujaen.es/recursos/spanish-wordnet-3-0/

Generalización de consultas web para protección de privacidad de usuarios

Estudio sobre la posibilidad de generalizar consultas web en un navegador para incrementar la privacidad de los usuarios

Silvia Cobo Gómez
Grado en Ingeniería Informática
Universitat Oberta de Catalunya
Palma de Mallorca, España
scobogo@uoc.edu

***Abstract*—E l trabajo propone estudiar la implementación de un mecanismo para generalizar consultas web que opere en modo local, es decir, que se instale en el lado del usuario.**

Para ello, se utilizará la herramienta *Wordnet* como base de conocimiento para la generalización de las consultas web, aunque de una forma reducida para no incurrir en problemas de espacio, almacenamiento o rendimiento en las búsquedas web.

Se evaluará, junto a una serie de usuarios voluntarios, el rendimiento de la aplicación, la privacidad que los usuarios perciben, el consumo de CPU memoria y el tiempo de procesamiento, para determinar la viabilidad y funcionalidad de esta.

In this project we propose to study the implementation of a tool that aims to generalize web search queries. This tool will operate locally, that is, on the user side.

For this, the Wordnet tool will be used as a knowledge base for the generalization of web queries, although in a reduced way so as not to incur space, storage or performance problems in web searches.

The performance of the application, the privacy perceived by users, the consumption of CPU memory and the processing time will be evaluated, together with a series of voluntary users, to determine its viability and functionality.

***Keywords*— Seguridad; privacidad; web; plugin**

I. PROBLEMA POR RESOLVER

En la actualidad, millones de usuarios utilizan a diario buscadores web para obtener información sobre cualquier asunto por el que sientan interés o sobre el que necesiten obtener más información. Para realizar estas búsquedas, los usuarios utilizan motores de búsqueda en sus navegadores de confianza. Motores tales como Bing, Yahoo! Search, etcétera. Estos motores de búsqueda pueden crear un perfil de usuario para ser capaces de, basándose en búsquedas anteriores del propio usuario, ofrecer los resultados más precisos al usuario en cuestión. Por ejemplo, si un usuario realiza a menudo consultas sobre planetas o el espacio y busca la palabra Mercurio, los resultados referentes al planeta aparecerán en primer lugar, sobre aquellos que se refieran al elemento químico. Este perfil permite que la experiencia de usuario sea más satisfactoria, dado que permite que las búsquedas ofrezcan el resultado esperado con más rapidez.

No obstante, la creación y utilización de estos perfiles de usuario plantea un problema de privacidad. Nuestro perfil de usuario, en definitiva, nuestros datos, están disponibles para el buscador en Internet, en su base de datos. ¿Qué pasaría si algún usuario malintencionado pudiera acceder a estos datos y, de alguna manera, acabar adivinando nuestra identidad real? ¿Existe alguna forma de proteger nuestra privacidad y, a la vez, seguir obteniendo resultados satisfactorios en las búsquedas?

En el presente trabajo se propone estudiar la implementación de un mecanismo para generalizar consultas web que opere en modo local, es decir, que se instale en el lado del usuario. La aplicación realizará una generalización sobre el término que el usuario desea buscar antes de que el usuario ejecute la búsqueda, y la sustituirá por el término generalizado

en el buscador. De este modo, se pretende proteger la privacidad del usuario con respecto a la búsqueda deseada, al haber sido modificada previamente. La generalización de la palabra permitirá obtener resultados semánticamente similares, pero con una capa de protección de la privacidad extra para el usuario. En concreto, en este trabajo se permitirá proteger la privacidad en el buscador web de Google en el navegador Google Chrome.

Se utilizará la herramienta Wordnet como base de conocimiento, para permitir la generalización de las consultas web del usuario.

Se evaluará, junto a una serie de usuarios voluntarios, el rendimiento de la aplicación, la privacidad que los usuarios perciben, el consumo de CPU memoria y el tiempo de procesamiento, para determinar la viabilidad y funcionalidad de esta.

II. Estado del arte

En la literatura, observamos que ya existen algunos métodos de generalización y falsificación de consultas [1] que permiten reforzar la privacidad de los perfiles de usuario en búsquedas web. Por ejemplo, existen métodos para falsificar las consultas en base a la información obtenida en el perfil de alguna red social del mismo usuario, por ejemplo, Twitter. Utilizando los datos de Twitter se crea una base de conocimiento del propio usuario que permite falsificar las consultas web con datos lo más similares posibles y adaptados al perfil del usuario, pero que no permitirían que un atacante que accede a ese perfil de usuario pudiera encontrar su identidad real en base a los datos obtenidos del perfil.

A pesar de existir algunas ideas y herramientas que explotan diferentes posibilidades de privacidad [2][3], ninguna de ellas se ha planteado hasta el momento el instalar en el lado del usuario esta herramienta, en forma de plugin en el navegador web. Por lo tanto, esto es lo que vamos a estudiar en este trabajo.

III. Objetivos generales

El proyecto consiste en estudiar e implementar una forma de generalizar consultas web para proteger la privacidad del usuario que realiza estas búsquedas y, posteriormente, evaluar la viabilidad de la herramienta desarrollada.

En concreto, esta forma de generalizar las consultas se llevará a cabo utilizando un plugin para el navegador Chrome, instalado en la parte del cliente, es decir, en la máquina del usuario que realiza las consultas.

Los objetivos generales, son:

Programar un plugin o aplicación que permita administrar el nivel de protección de la privacidad del usuario en cuanto a consultas web, aportando diferentes niveles de privacidad a su perfil.

No debe ser necesario ampliar el hardware de la máquina del cliente, por ello todo el peso de la operativa debe realizarse en la nube o en infraestructura externa, propia de la aplicación.

El plugin debe instalarse en el lado del cliente, es decir, en su propio navegador.

La utilización plugin no debería afectar al rendimiento de la máquina del cliente ni aumentar excesivamente el tiempo de búsqueda de los términos en el buscador.

Evaluar la viabilidad de esta aplicación utilizando una batería de pruebas con la colaboración de varios usuarios de prueba, que nos permitirán detectar los puntos fuertes y débiles de la misma.

Recoger los comentarios de usuarios y los propios para elaborar unas conclusiones y ofrecer una idea de versión mejorada teniendo en cuenta todos los puntos anteriores.

IV. METODOLOGÍA Y PROCESO DE TRABAJO

La herramienta que vamos a utilizar como base para nuestro proyecto es WordNet. Se trata de una base de datos que contiene palabras, sus definiciones y sus relaciones semánticas. Al ser un producto final ya disponible para su descarga y utilización, partimos de una base sólida para comenzar todo el trabajo de desarrollo a su alrededor.

Wordnet nos permitirá generalizar las búsquedas realizadas en la web, para adecuar el nivel de privacidad a elección del usuario. Por ejemplo, nos permitiría generalizar la búsqueda de alguna enfermedad o trastorno, como podría ser el TOC (trastorno obsesivo-compulsivo) a "enfermedad mental" o, incluso, a enfermedad simplemente, dependiendo del nivel de privacidad indicado.

Tras haber analizado Wordnet, decidimos establecer dos niveles de privacidad para el usuario: uno más simple, para enmascarar ligeramente la consulta, y otro que ofrezca un nivel máximo de privacidad (a pesar de salir perdiendo en cuanto a lo acertado de los resultados).

Se realizará una función en lenguaje Python para consultar la base de datos de Wordnet. El programa recogerá los términos de la búsqueda realizará por el usuario y, aplicando los niveles de privacidad requeridos, cambiará los mismos para regenerar la consulta que se va a enviar a Google.

La prueba de concepto del programa consistirá en programar un plugin para el navegador web Google Chrome.
Una vez elaborada la primera versión de este, se concertarán entrevistas con varios usuarios voluntarios que nos permitirán instalar el software en sus máquinas y realizarán una serie de pruebas preestablecidas para evaluar el funcionamiento y rendimiento de la herramienta. Una vez realizadas las pruebas preestablecidas, se les dejará también realizar pruebas a libre albedrío.

Una vez tengamos las conclusiones por parte de los usuarios, pasaremos a finalizar la memoria con las conclusiones del proyecto, recogiendo los puntos fuertes, los que se podrían mejorar, lecciones aprendidas, etc.

V. WORDNET

Wordnet es un proyecto creado en 1985 [4] y que sigue siendo mantenido desde entonces por la Universidad de Princeton, en concreto por el 'Cognitive Science Laboratory'. Consiste en una base de datos léxica del idioma inglés, cuya descarga se ofrece de forma abierta y gratuita a los usuarios que deseen utilizarlo.

El propósito de Wordnet consiste en la agrupación de sustantivos, adjetivos, etc. en conjuntos de sinónimos cognitivos, que ellos denominan synsets. Los synsets se conectan entre ellos por medio de relaciones conceptuales-semánticas y léxicas, dando como resultado una red de palabras y conceptos con aplicaciones diversas.

Entre estas aplicaciones se encuentran clasificaciones automáticas de diferentes textos, traducciones automáticas, generación de crucigramas, resúmenes de texto, etc.

En nuestro caso, utilizaremos la base de datos de generalización de las palabras, también conocido como hiperónimos. Buscaremos el synset de la palabra que se desea buscar y obtendremos el hiperónimo más adecuado siguiendo ciertas reglas que se han estudiado previamente.

VI. Función para generalizar consultas

Para esta prueba de concepto, nos hemos centrado en búsquedas de una sola palabra, es decir, no se realizan búsquedas de frases. A pesar de que este proceso se puede paralelizar para permitir generalizar la búsqueda de una frase, la generalización de palabras separadas puede alejarse demasiado del término inicial que se deseaba buscar. Además, el tiempo incrementaría a la hora de montar la respuesta para poder enviar la búsqueda a Google.

Utilizaremos los hiperónimos de Wordnet para generalizar las consultas realizadas por los usuarios, permitiendo generalizar en un nivel básico y en un nivel más restrictivo.

Se han tomado varias decisiones para agilizar y simplificar el proceso.

1- Simplificación de consultas: Cuando se realiza una consulta a Wordnet, aparecen resultados para todos los posibles significados de esa palabra. Como no tenemos ninguna información sobre el perfil de usuario en nuestra aplicación (de hecho, es lo que queremos evitar tener), no podemos saber exactamente a cuál de los significados se refiere. Por lo tanto, asumiremos que el primero de todos los resultados devueltos es el correcto.

Si realizamos una consulta buscando aids que serían las siglas en inglés para la enfermedad del SIDA, el primer resultado se refiere a la enfermedad, que es lo que estamos buscando. El resto de los resultados se refieren a otros significados tales como trabajo, persona, soporte… que podrían ser el objetivo de búsqueda de algún usuario pero que por regla general no será así.

2- Agilizar la generalización: Una vez elegido el significado, podemos obtener un listado de hiperónimos, es decir, de generalizaciones para el mismo. A veces, esta búsqueda devuelve un único significado y otras devuelve un listado.

Tras un estudio previo, se ha decidido que la primera palabra de la lista se corresponde con un nivel bajo de generalización y, así como leemos los resultados que va ofreciendo, la generalización es cada vez mayor.

Por ejemplo, siguiendo con el ejemplo del SIDA, el primer nivel de generalización devolvería una consulta a “inmunodeficiencia” y el nivel más restrictivo “enfermedad infecciosa”.

Por el momento no se ha modificado la base de datos de Wordnet para reducirla y así reducir también el tiempo de respuesta, puesto que la complejidad de la ejecución de la tarea no compensa la ganancia de tiempo de consulta y apenas añade unas décimas de segundo a la búsqueda.

El código implementado es bastante sencillo, y únicamente ejecuta los puntos comentados anteriormente: la función recibe la palabra a generalizar y el nivel de generalización deseado.

```
    Defgeneralize_search(word, basicPrivacy):
if wn.synsets(word):
  results = [lemma.name() for synset in wn.synsets(word)[0].hypernyms()
   for lemma in synset.lemmas()]
  if basicPrivacy:
    return results[0].replace("_", " ")
  else:
    return results[len(results) - 1].replace("_", " ")
return word
```

Con estos datos, busca la generalización y devuelve la palabra generalizada adecuada. En el caso de no existir la palabra en la base de datos de Wordnet, devuelve la misma palabra a buscar sin modificar, para que el usuario pueda realizar la búsqueda sin perder más tiempo.

En resumen, la función realiza los pasos contenidos en el siguiente diagrama:

- Recoge como input la palabra
- Realiza la consulta a Wordnet

- Recoge el primer significante de la palabra
- Obtiene la primera y última generalización de la palabra en cuestión

Consulta si el usuario deseaba la generalización básica o estricta

Devuelve la generalización solicitada, si está disponible

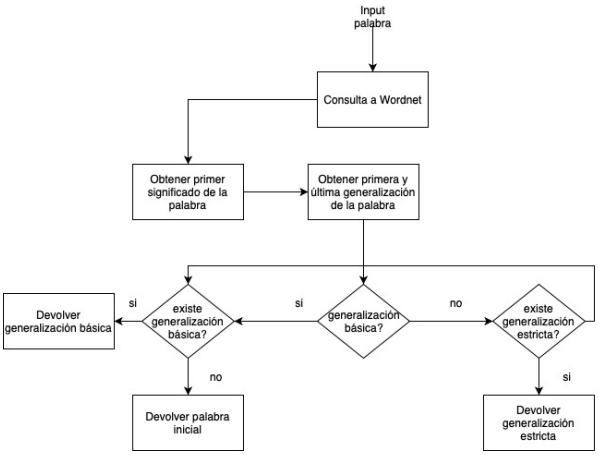


Figure 1: Diagrama flujo función python

VII. EXTENSIÓN GOOGLE CHROME

Como habíamos comentado anteriormente, debemos proporcionar algún tipo de herramienta para que el usuario final pueda realizar sus pruebas en la herramienta desarrollada. En este caso, hemos decidido realizar una extensión instalable en el navegador Google Chrome.

Estas extensiones permiten ejecutar funciones y programas predeterminados en el navegador Google. También se puede elegir el momento de ejecución, por ejemplo, al abrir una nueva pestaña, al cerrarla, al ejecutar una búsqueda…

Las extensiones de Google Chrome consisten en una carpeta en la que se encuentran diferentes archivos. Uno de ellos es el archivo manifest.json, donde se incluye la información básica de la extensión o plugin y los permisos necesarios, el momento en el que se tiene que ejecutar el plugin, etc.

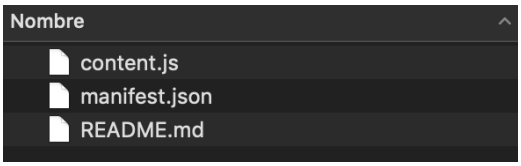


Figure 2: Estructura de carpetas plugin

El fichero más importante es el fichero de contenido (content.js). Consiste en un código javascript que contiene los pasos a ejecutar para ejecutar la funcionalidad deseada. En nuestro caso, capturar el texto del buscador, consultar la función que, a su vez, consulta Wordnet, y devolver el resultado a google justo antes de ejecutar la búsqueda, para que así se sustituya por el término generalizado.

Actualmente el código javascript recoge cualquier cambio que se realiza en el cuadro de texto del buscador de google, envía esta palabra a un servidor que se conecta a la función Python que comentábamos en el apartado anterior, y ésta devuelve la generalización de la palabra, según el nivel de privacidad deseado.

Para consultar la función Python encargada de devolver la generalización, utilizamos un servidor web ubicado en la nube, en concreto en aws. Este servidor alberga el código Python y un servidor web que escucha en el puerto 8787 las consultas, y ofrece el resultado de estas.

La infraestructura del servidor aws consta internamente de:

Un servidor https para servir las consultas al código Python. Para ello utiliza un nginx como primer punto de contacto y receptor de las consultas. Es en este punto donde hemos creado un certificado auto-firmado para poder cumplir los estándares de Google y ofrecer protección ssl en las consultas.

Un servidor web https que recoge la información recibida vía nginx mediante un puerto distinto y que, a su vez, consulta la función Python

La función Python en sí, que ya se ha explicado en apartados anteriores, que recoge palabra y nivel de privacidad, y devuelve la información correspondiente.

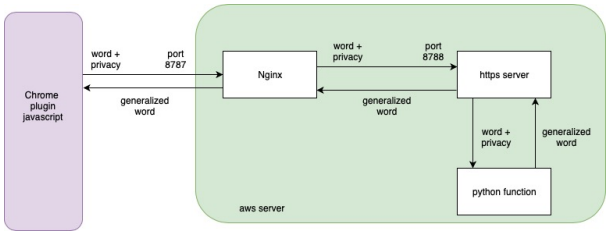


Figure 3: Infraestructura servidor aws

El nivel de privacidad de la consulta se establece en el propio plugin, concretamente en el fichero content.js, en la línea que contiene la URL del servidor web. Si escribimos True, tenemos privacidad básica y, en caso de desear una privacidad más estricta, podemos indicar False:

xhttp.open("GET", "https://35.180.253.208:8787/"+this.value+"/**True**", false);

La función Python a la que se llama, recibe la palabra y el nivel de privacidad y devuelve la generalización solicitada en base a esos parámetros.

Se contempla el caso de que sólo devuelva un resultado de generalización y también se contempla el caso en el que la palabra no exista en la base de datos. En ese caso, se devuelve como resultado la palabra original.

El nivel de privacidad se debe establecer en el código del plugin, y se puede modificar directamente en el ordenador de cada uno de los participantes en el estudio. Se les permite cambiar el nivel de privacidad accediendo al código del plugin y recargándolo ellos mismos, puesto que es una tarea sencilla. No obstante, se ha considerado fuera del alcance las pruebas con usuarios y se ha ejecutado por mi tras una semana de pruebas en cada modo.

VIII. EL PRODUCTO FINAL

El resultado final de esta prueba de concepto es una extensión de google Chrome que utiliza un algoritmo basado en Wornet para permitir la generalización de la consulta en base a los parámetros dados y la palabra deseada. El funcionamiento a nivel lógico del aplicativo puede verse en el siguiente diagrama:

El usuario introduce su búsqueda en el buscador del navegador web Google Chrome

El plugin entra en acción y ejecuta su secuencia

Llama al servidor web para obtener la generalización solicitada

El servidor web provee el listado de resultados

Se recorre este listado para descartar los resultados no deseados: si se desea aplicar privacidad básica, se descartan los resultados de privacidad estricta, etc.

Si la generalización deseada está disponible, el plugin devuelve esta palabra y ésta se busca en Google

En el caso de que la generalización deseada no esté disponible, se ofrecerá la siguiente alternativa disponible. Es decir, si se desea privacidad estricta pero solo se encuentra privacidad básica, se devolverá esta última. Si tampoco se obtiene generalización básica, se devuelve la búsqueda inicial.

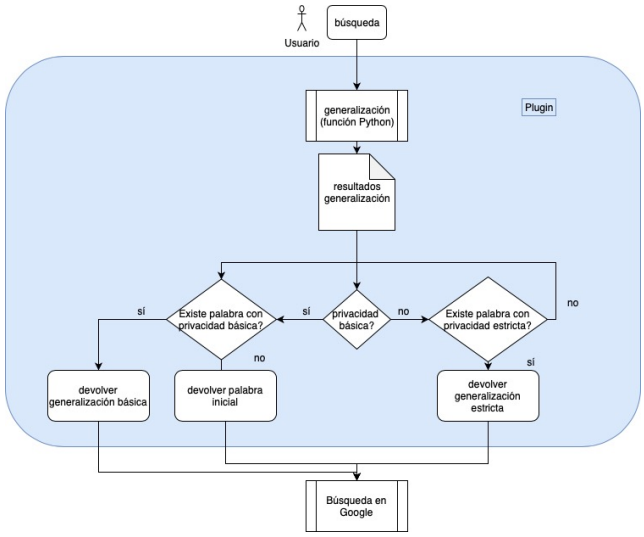


Figure 4: Diagrama flujo plugin

Por lo tanto, en el proceso de búsqueda, la extensión se encarga de realizar una modificación en el momento en que el usuario pulsa el botón de búsqueda y cambia la palabra a

buscar por la generalización de esta, en el caso de que esta exista.

IX. PRUEBAS CON USUARIOS

Hemos contado con la colaboración de los siguientes perfiles y usuarios:

Perfiles técnicos:

- Oscar Cobo Gómez, programador Java
- José Laredo López, ingeniero en informática de sistemas
- Daniel Nieto, técnico informático

Perfiles no técnicos:

- Ana María Gómez Lázaro, camarera de piso
- Delia Martínez Flores, quiromasajista

Los ordenadores utilizados para las pruebas son ordenadores de mesa básicos o portátiles estilo netbook, es decir, que no poseen grandes prestaciones técnicas. El uso de este tipo de máquinas no debería penalizar en ningún caso el uso del plugin, puesto que este se centra en el uso web y en el servidor que realiza las consultas a Wordnet que se encuentra en la nube.

Ambos grupos de usuarios han realizado una batería de búsquedas con el plugin en modo privacidad básica y privacidad estricta. Las pruebas consistían en realizar una serie de consultas sobre diversas enfermedades en google y evaluar la rapidez en la ejecución de estas (1-5) y la satisfacción con los resultados obtenidos (1-5). Seguidamente, también se les pide que utilicen el plugin libremente para obtener sus impresiones

- Privacidad básica: Los usuarios otorgan una media de 5/5 en rapidez en las búsquedas y una media de 3’66/5 en la precisión de las búsquedas

La media de precisión los usuarios técnicos es de 4’1/5 y de los usuarios no técnicos es de 3’05/5. Por lo tanto, los usuarios técnicos han obtenido resultados más satisfactorios que los usuarios no técnicos.

-Privacidad estricta: Los usuarios otorgan una media de 5/5 en rapidez en las búsquedas y una media de 3’3/5 en la precisión de las búsquedas.

La media de precisión los usuarios técnicos es de 3’73/5 y de los usuarios no técnicos es de 2’65/5. Por lo tanto, los usuarios técnicos han obtenido resultados más satisfactorios que los usuarios no técnicos, al igual que en el caso anterior, con las búsquedas en modo privacidad básica.

Comparativa satisfacción según privacidades

Tras revisar los resultados del apartado anterior y ver que todos los usuarios nos han dado la nota máxima en rapidez de ejecución de las consultas, vamos a pasar a analizar únicamente las satisfacciones con respecto al resultado obtenido.

En este caso, podemos ver que hay discrepancias entre las opiniones de los diferentes usuarios, no obstante, sus resultados son concordantes con el resto de los usuarios de su mismo perfil así que procedemos a agruparlos en el siguiente gráfico:

Figure 5: Comparativa usuarios

En ambos casos, los usuarios técnicos ofrecen puntuaciones más altas, tanto para privacidad básica como para privacidad estricta, y los usuarios de perfil no técnico puntúan más bajo. No obstante, a pesar de las puntuaciones más bajas, el resultado medio sigue siendo una nota de aprobado (superior a 2’5 sobre los 5 puntos totales asignables)

Conclusiones test con usuarios

Según las puntuaciones asignadas por los usuarios, obtenemos las siguientes conclusiones:

Los usuarios de ambos perfiles consideran que los resultados aparecen con bastante rapidez, es decir, que no se añade tiempo de espera excesivo al utilizar el plugin. Indican que no encuentran apenas diferencia entre una búsqueda normal y una búsqueda con el plugin. Por lo tanto, parece que cumplimos con el objetivo de no añadir tiempo ni consumir recursos de la máquina del cliente.

Los resultados son más satisfactorios con el filtro de privacidad básico, como era de esperar. No obstante, en algunas pruebas, algunos usuarios consideran que la búsqueda con privacidad estricta cumple mejor sus expectativas con respecto a la búsqueda con privacidad básica. En este caso, esto también podría ser predecible, puesto que no todos los usuarios tienen el mismo tipo de expectativas con respecto a una palabra que el resto, y la generalización estricta puede acercarse más al concepto que estos tienen de la palabra.

En general, los usuarios con perfil técnico están más satisfechos con los resultados que los usuarios con perfil no técnico. Esto puede deberse a que, al estar familiarizados con lo que se ha hecho en el plugin, la tecnología utilizada y sus restricciones, sean más comprensivos con el proceso que sigue el programa.

Los comentarios recogidos en cuanto a las pruebas libres se refieren a que es complicado hacer vida normal con este plugin, dado que los resultados siempre salen algo desvirtuados y les dificulta encontrar lo que buscaban en un inicio, haciéndoles invertir más tiempo del necesario e incluso teniendo que cambiar de navegador a otro sin el plugin para obtener los resultados que desean, porque no quieren buscar las palabras en inglés. Además, el problema de no poder buscar frases completas resta más puntos a la experiencia de usuario.

Los perfiles no técnicos concluyen en que no lo utilizarían en su día a día. Les resulta poco útil y no están acostumbrados a tener que cambiar el idioma a inglés para sus búsquedas. Como ya hemos mencionado anteriormente, el problema de no poder buscar frases también les dificulta el uso habitual en su día a día.

Los usuarios de perfil técnico coinciden en que lo utilizarían en su ordenador privado cuando se incluyera la posibilidad de buscar frases pero que, para sus búsquedas profesionales, referidas a términos técnicos de programación/sistemas, no ofrece un uso satisfactorio. Les ha gustado mucho como funciona el plugin y estarían dispuestos a seguir participando en pruebas si se quisiera continuar el desarrollo.

X. CONCLUSIÓN

Tras realizar la programación de la prueba de concepto del plugin y las pruebas con usuarios, podemos concluir que la idea estudiada en este trabajo constituye una buena base para seguir trabajando en el futuro.

Los resultados son buenos: el tiempo de consulta para la generalización es prácticamente despreciable y, en general, la satisfacción de los usuarios con los resultados obtenidos es bueno. No obstante, se recogen una serie de puntos de mejora de cara a una segunda versión del plugin.

Los primeros puntos de mejora detectados se centran en la herramienta Wordnet en sí. La primera limitación detectada es la barrera del idioma, puesto que actualmente sólo se permite realizar consultas en inglés. Si bien se han encontrado algunas referencias a adaptaciones de la base de datos de Wordnet al español [5], en el momento no existe ninguna alternativa de código libre disponible. Por lo tanto, centrarse en encontrar o elaborar una adaptación de la base de datos de Wordnet al español podría ser un primer paso para mejorar el trabajo realizado.

Por otra parte, siguiendo con Wordnet, la siguiente limitación se refiere a la capacidad de poder consultar frases

completas o varias palabras. En este momento, únicamente se pueden buscar palabras por separado. Se debería modificar el plugin para permitir generalizar más palabras en la misma consulta, a pesar de que esto suponga realizar varias consultas a Wordnet. Un problema derivado de esto es la posible desvirtuación de la búsqueda, al generalizar cada una de las palabras por separado, la probabilidad de que el resultado no tenga sentido una vez generalizado es muy alta. Por lo tanto, es un apartado a estudiar en otra prueba de concepto, para evaluar la mejor forma de salvaguardar este obstáculo.

En tercer lugar, también relacionado con Wordnet, tenemos un problema relacionado con el listado de hiperónimos, es decir, de las generalizaciones. Hemos asumido que la primera palabra que devuelve el listado es la generalización básica y que la última sería la de mayor generalización, pero no ocurre así en todos los casos, puesto que asumimos un significado de la palabra que puede no ser el que quiera el usuario. Se debería rehacer un estudio de este listado y, además, tal vez ofrecer una generalización de más niveles.

Relacionado también con el punto anterior, hay muchas palabras que únicamente devuelven una palabra en el listado de generalizaciones. Por lo tanto, otro punto de mejora a añadir sería el de revisar la base de datos para poder ampliarla con más información. Al ser una tarea que debe hacerse a mano directamente en la base de datos descargada en local, podría ser una tarea muy tediosa. Como hemos comentado anteriormente, se debería revisar la base de datos de Wordnet para adaptarla al idioma español y, de paso, podríamos preparar un acceso mediante api o similar, para poder modificar la base de datos de palabras mediante alguna web o llamada a este api, para así poder añadir o borrar información. Si exportamos la base de datos a mysql, podríamos acceder por un simple portal web para añadir las informaciones que fueran necesarias de una forma más cómoda.

Para acabar, en este momento tenemos una función Python que se ejecuta en un servidor web en la nube, concretamente en la nube de Amazon (AWS). La máquina no precisa de una gran infraestructura, por lo tanto, de momento su uso es gratuito, es decir, no nos incurre en ningún tipo de coste. Pero, si queremos modificar la base de datos, deberíamos buscar donde alojarla y, en este caso, si será necesario invertir algo de presupuesto en la infraestructura necesaria.

El estudio y la realización de estas mejoras, y su posterior evaluación con usuarios, supondría un avance tal que la aplicación o plugin podría ofrecerse al mercado de plugins, para así ser utilizado por usuarios finales alrededor del mundo, y que este se introdujera en el día a día de muchos usuarios que deseen proteger su privacidad.

XI. BIBLIOGRAFÍA

[1] Javier Parra-Arnau, David Rebollo-Monedero, Jordi Forné, "Optimal Forgery and Suppression of Ratings for Privacy Enhancement in Recommendation Systems," (MDPI) Entropy, vol. 16, no. 3, Mar. 2014, pp. 1586-1634. DOI: 10.3390/e16031586.

[2] Alexandre Viejo, David Sánchez, "Profiling Social Networks to Provide Useful and Privacy-Preserving Web Search", Journal of the Association for Information Science and Technology, vol. 65, no. 12, pp. 2444-2458, 2014.

[3] David Sánchez, Jordi Castellà-Roca, Alexandre Viejo, "Knowledge-Based Scheme to Create Privacy-Preserving but Semantically-Related Queries for Web Search Engines", Information Sciences, vol. 218, pp. 17-30, 2013.

[4] Princeton University, “Wordnet, A Lexical Database for English”, Recuperado de: https://wordnet.princeton.edu

[5] Ana Fernández, Glòria Vázquez, Irene Castellón. (2008). Spanish WordNet 3.0. Spain. Red Temática en Tratamiento de la Información Multilingüe y Multimodal (TIMM). Recuperado de: http://timm.ujaen.es/recursos/spanish-wordnet-3-0/