Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Computación

“Sistema de gestión del conocimiento para el acompañamiento en el estudio de la gramática para la presentación de la certificación TOEFL (SG TOEFL)”

T E S I S

Para obtener el grado de:

Licenciatura en ciencias de la computación

Presenta:
Aaron Ramírez Martínez

Asesora:
Dra. Meliza Contreras González

24 de mayo de 2024

Contenido

Resumen

Esta tesis describe un sistema para la interpretación y procesamiento de la información de bancos de información para la certificación TOEFL. Se propone un método que procesa información de un archivo de texto y como salida muestra una nube de palabras más comunes dentro del texto, el texto es procesado con librerías de procesamiento natural del lenguaje para poder determinar que palabras son conectores, preposiciones y cuales no, pues estas no aportan un contenido altamente relevante al documento.

La certificación TOEFL requiere una gran capacidad de conocimientos en el idioma, por lo que es muy importante la preparación para presentar dicha certificación, sí bien existen muchos exámenes en línea que sirven para medir el nivel de conocimientos para la certificación, no existen muchas herramientas para ejemplificarnos cual es el principal tema a tratar en la certificación.

Introducción

Descripción del problema

La pandemia por SARS-COVID nos ha traído consecuencias negativas en materia de aprendizaje, puesto que muchas de las instituciones que tenían por bien preparar a los aspirantes a la aplicación del examen han tenido que migrar a plataformas digitales que puedan facilitar el aprendizaje sin comprometer la integridad física de las personas.

Es por ello que surge la idea de SG TOEFL para que permita entrenar a los aspirantes para la aplicación del examen, para lograr una mejora en la calidad de los resultados de las aplicaciones que sean acompañados de esta herramienta.

La idea surge a partir de experiencias personales, puesto que el nivel de certificación TOEFL requiere una gran capacidad de conocimiento y experiencia en el manejo del nivel de idioma inglés, en busca de herramientas que permitan enriquecer el conocimiento, no existe una gran variedad, sí bien existen plataformas de entrenamiento como un simulacro de examen y preguntas tanto de manera escrita como de manera auditiva, de las cuales abundan; no existe una gran variedad de herramientas que en forma didáctica puedan enseñar los tópicos principales del examen.

Es por ello que el presente proyecto de tesis tiene como principal objetivo crear una herramienta que haga uso de un banco de información, y a su vez genere en los usuarios una alternativa de estudio y entrenamiento de la gramática para la aplicación del examen.

Motivación

El poder del procesamiento de la información del lenguaje Python hoy en día es muy grande, así como el procesamiento de imágenes y de su contenido, han crecido de manera exponencial en los últimos años, es por ello que la idea nace a partir el poder de las herramientas y la poca información que con ella es procesada para necesidades académicas, pues el idioma ha sido un factor importante por el que muchas personas o profesionistas aspirar a tener un empleo mejor remunerado.

Un gran aumento en la tasa de crecimiento de profesionales egresados ha sido una clave importante para el desarrollo de esta herramienta.

Por otro lado, en procesamiento del lenguaje natural y el rápido crecimiento de la recuperación de la información dan pie al desarrollo de esta herramienta, procesando texto y haciendo un análisis de la información.

Este trabajo presenta sistema en lenguaje Python que procesa la información brindada en un ***corpus*** en formato txt, y extrae la información más relevante, que devuelve en otro documento txt con el **corpus** de salida correspondiente, a partir

del cual podremos ver de manera gráfica cuales son las principales palabras usadas en el **corpus**.

Objetivo general

Generar un Sistema de gestión del conocimiento para el acompañamiento en el estudio de la gramática para la presentación de la certificación TOEFL(SG TOEFL).

Objetivos específicos

Creación de un algoritmo de relación de gramática en recuperación de texto.

Diseño y maquetación del repositorio de información, así como su preprocesamiento de los textos.

Estructura de la tesis

El contenido de esta tesis se organiza de la siguiente manera; En el capítulo 1 se describen los conceptos básicos, en el capítulo 2 se describen los tipos de procesos de recuperación de información y clasificación de los mismos, en el capítulo 3 se presenta el algoritmo de PageRank que funciona como base de nuestra clasificación para la implementación, pero que es relevante para nuestra tesis, en el capítulo 4 se definen los conceptos previos y la conceptualización de nuestro algoritmo, en el capítulo 5 se muestran los resultados cuantitativos del desarrollo de nuestro proyecto.

Capítulo 1 | Conceptos previos

El articulo *EL uso de apps para el aprendizaje del inglés a través del modelo B-Learning* nos da un panorama amplio acerca del fortalecimiento de las capacidades de comunicación en el idioma inglés (*Listening, Writing, Speaking, Reading)* y cuya principal base de estudios es la premisa de cómo es que el uso de aplicaciones móviles favorece en el proceso de aprendizaje [1].

En la tesis de maestría titulada *"Propuesta de taller sobre estrategias para mejorar la comprensión auditiva en inglés para estudiantes de lenguas extranjeras"*, el estudio semiestructurado de la FGU-BUAP nos informa sobre posibles formas de incidir y corregir en el aprendizaje de un grupo de personas que hablan una lengua extranjera y, de hecho, el tema central es que el taller se lleva a cabo de manera física, pero sienta las bases para algunas de las preguntas de aprendizaje que coinciden con el método de llevar a cabo el proyecto [2].

De la tesis de maestría *MOODLE como herramienta para el aprendizaje de inglés como lengua extranjera: un estudio de caso* observamos un análisis cuantitativo y específicos de las causas y consecuencias del estudio en plataformas digitales en contextos universitarios como APOYO, en procesos de enseñanza, y aprendizaje de lenguas extranjeras [3].

Sí bien, existen muchos exámenes simuladores en internet cuyo fin es "entrenar" al usuario para su examen de certificación TOFEL mucho de estos reactivos son simplemente tomados de las guías de estudio que existen en la red y puestas en un test cronometrado. La finalidad no es suplantar ninguna herramienta existente en línea.

1.1 Recuperación de la información

Recuperar significa volver a tener, la recuperación de la información (RI) es una disciplina que estudia la representación, la organización y el acceso eficiente a la información que se encuentra registrada.

Ahora bien, para que la RI tenga sentido se presupone un entorno en el cual no es trivial, precisamente, el hecho de acceder a los documentos por su contenido. Este contexto lo genera, típicamente, cualquier fondo documental a partir del momento que contenga unos centenares o unos miles de documentos. Empresas pequeñas, medianas o grandes, con ejecutivos, abogados, químicos o ingenieros que necesitan encontrar una información en fondos internos o externos es un ejemplo. Universitarios e investigadores que necesitan consultar bases de datos bibliográficas para asegurarse de que no reinventan la rueda es otro. Finalmente, la Web, que en realidad es un enorme sistema de información documental con varios miles de millones de documentos es el ejemplo extremo de contexto característico de RI.

Existen rasgos característicos de los sistemas de RI, por ejemplo.

Uno: Aunque su principal eje central de desarrollo radica en los ordenadores, como muchos otros sistemas de información actual, varían mucho, algunos son asistidos por computadora y otros mediante ordenadores.

Dos: Gestionar información de cualquier tipo, desde texto hasta elementos multimedia, como fotos o videos, pero siempre usando información textual.

Uso de ordenadores: La RI tiene como base el uso de ordenadores y, por lo tanto, el manejo de grandes cantidades de información o bases de datos o sistemas automatizados de procesamiento de información como por ejemplo los hipertextos.

El uso de la información textual gestiona información textual de tipo prosa, en lugar de, por ejemplo, datos numéricos o alfanuméricos, mejor estructurados, como lo hacen ya otros sistemas, cuando la RI gestiona documentos u objetos no textuales, como archivos multimedia, lo hace a través de los metadatos propios de la información.

El desarrollo de herramientas en sistemas de información, son muy poco prácticos hoy día, por ejemplo, un sistema de información documental mediante el uso de una base de datos relacional y normalizada, no podrá satisfacer de manera adecuada la necesidad de *descubrir* información sí no, solo ampliarla.

La utilidad principal de un sistema de este tipo será parcial, porque su propósito principal es extender la información que se tiene y no ampliarla de manera asertiva y eficaz como se espera, por otro lado, un buen sistema de RI se espera pueda satisfacer las necesidades nuevas de descubrimiento.

1.2 Corpus

Un corpus en el ámbito de la recuperación de información constituye un conjunto meticulosamente seleccionado y debidamente etiquetado de documentos, empleado con el propósito de evaluar y avanzar en los sistemas relacionados con dicha recuperación. Este conjunto de datos provee la base necesaria para la evaluación y comparación de tales sistemas, siendo seleccionados y etiquetados con el fin de cumplir rigurosamente con criterios predefinidos como la relevancia, calidad y diversidad. Así, se otorga a los investigadores la capacidad de someter a los sistemas de recuperación de información a un ambiente que replica la realidad.

Los corpus resultan fundamentales en el proceso de evaluación de los sistemas de recuperación de información, ya que permiten la medición tanto de su precisión como de su eficiencia. La precisión alude a la habilidad de un sistema para recuperar documentos pertinentes, mientras que la eficiencia se refiere a su capacidad para realizar dicha recuperación en un tiempo razonable.

Además de su función evaluativa, los corpus desempeñan un papel crucial en el desarrollo de los sistemas de recuperación de información. Constituyen una herramienta invaluable para entrenar y refinar los algoritmos asociados, con miras a mejorar tanto su precisión como su eficiencia.

En resumen, un corpus representa un compendio selecto y catalogado de documentos utilizado en la esfera de la recuperación de información, destinado a la evaluación y progreso de los sistemas pertinentes. Proporcionan un entorno controlado para la evaluación de la precisión y eficiencia de tales sistemas, a la vez que permiten a los investigadores desarrollar y optimizar los algoritmos inherentes a los sistemas de recuperación de información.

1.3 Representación gráfica de los resultados de la RI

La nube de palabras es una herramienta visual que se utiliza para representar la frecuencia de palabras en un texto o colección de documentos. Se utiliza comúnmente en la recuperación de información para representar la distribución de términos en un corpus de documentos, y para proporcionar una representación gráfica de los términos más importantes.

Una de las principales ventajas de las nubes de palabras es que permiten identificar rápidamente los términos más relevantes en un corpus de documentos. Al representar los términos con un tamaño proporcional a su frecuencia, es fácil para los usuarios identificar los términos más importantes y relevantes. Esto es especialmente útil en la exploración temática de un corpus de documentos, ya que ayuda a los usuarios a entender el contenido general del corpus y a formular preguntas de investigación.

Además, las nubes de palabras también proporcionan una visualización atractiva y fácil de comprender de los términos más importantes en un corpus de documentos. Esto las hace útiles para la comunicación de investigaciones y resultados a audiencias no técnicas. Por ejemplo, las nubes de palabras pueden ser utilizadas para mostrar los temas principales en una colección de noticias o en un conjunto de opiniones en una encuesta, de manera sencilla e intuitiva.

Otra ventaja de las nubes de palabras es su capacidad para identificar patrones y tendencias en los datos. Por ejemplo, las nubes de palabras pueden utilizarse para comparar la frecuencia de términos entre diferentes documentos o colecciones de documentos, lo que puede ayudar a los investigadores a identificar patrones y tendencias en la información.

En el ámbito empresarial, las nubes de palabras también son útiles para analizar las opiniones de los clientes a través de reseñas o comentarios en línea. Se pueden utilizar para identificar problemas comunes, tendencias y patrones en las opiniones de los clientes, lo que puede ayudar a las empresas a mejorar sus productos y servicios.

En resumen, las nubes de palabras son una herramienta valiosa en la recuperación de información ya que permiten identificar rápidamente los términos más relevantes en un corpus de documentos, proporcionan una visualización fácil de comprender, y ayudan al usuario final a vislumbrar la información de mejor manera.

1.4 Los metadatos en la recuperación de la información

Es claro que la elección de palabras para expresar una necesidad de información surge de un proceso de representación. Esta es una tarea intrincada que implica la utilización de los conocimientos y procesos cognitivos del usuario. Es relevante mencionar que , Luna-González (2015) menciona que “el trabajo de representación del conocimiento es el proceso que realiza el documentalista al mediar entre la información producto del conocimiento y el usuario final” (p. 78).

Los sistemas automatizados de RI con un enfoque cognitivo constituyen modelos que buscan parecerse al comportamiento humano desde las ciencias computacionales, bajo este contexto, existen diferentes vertientes para la representación de la información.

Los metadatos son datos que ofrecen detalles o suministran información acerca de otros datos. En el proceso de recuperación de información, los metadatos tienen un papel crucial al brindar detalles sobre los documentos o recursos informativos existentes, facilitando así a los usuarios la localización y acceso a la información requerida de manera eficiente.

Existen diferentes tipos de metadatos, pero algunos de los más comunes son los metadatos de descripción, los metadatos de indexación y los metadatos administrativos.

Los metadatos de descripción proporcionan información básica sobre un recurso de información, como su título, autor, fecha de publicación y resumen. Estos metadatos suelen ser utilizados para generar descripciones breves de los recursos de información, y son esenciales para que los usuarios puedan identificar rápidamente si un recurso es relevante para sus necesidades.

Los metadatos de indexación, por otro lado, proporcionan información detallada sobre un recurso de información, como palabras clave, términos de materia y categorías temáticas. Estos metadatos son utilizados para indexar y clasificar los recursos de información, lo que permite a los usuarios encontrar fácilmente información relacionada con un tema específico.

Los metadatos administrativos, por último, proporcionan información sobre la administración y gestión de un recurso de información, como quién lo creó, quién tiene permiso para accederlo y cuándo fue actualizado por última vez. Estos metadatos son esenciales para garantizar que los recursos de información estén disponibles y se utilicen de manera apropiada [11].

En resumen, los metadatos son una herramienta esencial en la recuperación de la información, ya que proporcionan información valiosa sobre los recursos de información disponibles, permitiendo a los usuarios encontrar y acceder fácilmente a la información que necesitan.

Además de estos tres tipos de metadatos mencionados, existen otros tipos como los metadatos de estructura, los metadatos de relaciones, los metadatos de seguridad, entre otros, cada uno con una función específica en la recuperación de la información y su uso dependerá del contexto y necesidad del sistema de recuperación de información.

Uno de los métodos más comunes de recuperación de información es la búsqueda full-text, que permite a los usuarios buscar información en el texto completo de los documentos almacenados en un sistema. Los sistemas de búsqueda full-text utilizan algoritmos de búsqueda e índices para encontrar documentos que contengan un término o conjunto de términos específicos.

La indexación y la clasificación de documentos son otras técnicas importantes utilizadas en los Sistemas de Recuperación de información para recuperar la información. La indexación es el proceso de asignar términos específicos a los documentos, mientras que la clasificación es el proceso de asignar categorías temáticas a los documentos. Juntas, estas técnicas permiten a los usuarios encontrar documentos relacionados con un tema específico.

Por último, los sistemas de recomendación de información utilizan algoritmos para sugerir documentos o recursos relevantes a un usuario en función de su historial de búsqueda y su comportamiento de lectura. Estos sistemas pueden ayudar a los usuarios a descubrir información nueva y relevante que de otra manera podrían haber perdido.

Los sistemas de recuperación de información son fundamentales para la eficacia de un sistema de gestión de información, ya que ayudan a los usuarios a encontrar la información que necesitan de manera rápida y eficiente. Además, estos sistemas deben ser fáciles de usar y personalizables, para que los usuarios puedan adaptarlos a sus necesidades específicas.

Es importante mencionar que la recuperación de información es un proceso continuo y en constante evolución, ya que las necesidades y expectativas de los usuarios y las tecnologías cambian con el tiempo. Por lo tanto, es importante que los sistemas de gestión de información se mantengan actualizados y adaptados a las nuevas tecnologías para garantizar que sigan siendo eficaces en la RI.

1.5 Metadatos

Los metadatos son una herramienta esencial en la recuperación de la información, ya que proporcionan información valiosa sobre los recursos de información disponibles, permitiendo a los usuarios encontrar y acceder fácilmente a la información que necesitan. Pero, ¿quién crea los metadatos?

La creación de metadatos es un proceso que involucra a varias partes interesadas. En primer lugar, los autores o creadores de los recursos de información, como los autores de libros, artículos y sitios web, proporcionan información básica sobre sus obras, como el título, el autor y la fecha de publicación. Esta información se utiliza para generar metadatos de descripción.

Además, los bibliotecarios y otros profesionales de la información proporcionan información adicional para los metadatos, especialmente para los metadatos de indexación y clasificación. Por ejemplo, los bibliotecarios utilizan términos de materia y palabras clave para describir los recursos de información, lo que permite a los usuarios encontrar fácilmente información relacionada con un tema específico.

Los administradores del sistema también juegan un papel importante en la creación de metadatos administrativos. Estos metadatos proporcionan información sobre la gestión y administración de los recursos de información, como quién tiene permiso para accederlo y cuándo fue actualizado por última vez.

Es importante destacar que la creación de metadatos no es un proceso estático, sino que es necesario actualizar y mantener los metadatos para que sigan siendo relevantes y precisos. También es importante que los metadatos sean interoperables, es decir, que se puedan compartir y utilizar entre diferentes sistemas de recuperación de información [13].

En resumen, la creación de metadatos es un proceso en el que participan varias partes interesadas, como los autores, bibliotecarios y administradores del sistema. Es importante que los metadatos sean precisos, actualizados y interoperables para garantizar que los usuarios puedan acceder fácilmente a la información que necesitan.

Capítulo 2 | Jerarquía de la información

2.1 Indexación de la información

La indexación de la información es un proceso mediante el cual se asignan términos específicos a los documentos, con el objetivo de permitir su recuperación y búsqueda de manera eficiente. La indexación de la información es esencial en los sistemas de gestión de información, ya que permite a los usuarios encontrar documentos relevantes en una gran cantidad de información almacenada.

Existen diferentes métodos de indexación, pero los más comunes son la indexación automática y la indexación manual. La indexación automática utiliza algoritmos y programas para analizar el contenido de los documentos y asignar términos específicos a los mismos. La indexación manual, por otro lado, requiere que un humano revise el contenido de los documentos y asigne términos específicos.

La indexación automática es más rápida y eficiente que la indexación manual, pero también tiene sus limitaciones. Los algoritmos de indexación automática pueden cometer errores y no siempre son capaces de asignar los términos adecuados a los documentos. Por otro lado, la indexación manual es más precisa, ya que los humanos son capaces de entender el contenido de los documentos de manera más profunda, pero es mucho más costosa y tardada.

Una vez que los documentos están indexados, se pueden utilizar los términos de indexación para buscar y recuperar información de manera eficiente. Los sistemas de búsqueda utilizan índices para almacenar los términos de indexación y asociarlos con los documentos correspondientes. Los usuarios pueden buscar información utilizando estos términos y los sistemas de búsqueda devolverán los documentos relevantes.

La indexación de la información también es esencial para la clasificación de documentos. La clasificación de documentos es el proceso de asignar categorías temáticas a los documentos, y se realiza utilizando los términos de indexación. Los documentos se pueden clasificar en categorías como ciencia, tecnología, medicina, entretenimiento, etc. Esto permite a los usuarios encontrar fácilmente información relacionada con un tema específico.

2.2 PageRank

El algoritmo de PageRank es un algoritmo de búsqueda desarrollado por Larry Page y Sergey Brin, los fundadores de Google, que es utilizado para determinar la importancia de una página web en relación con otras páginas en internet. El algoritmo es la base de la tecnología de búsqueda de Google y ha sido utilizado para clasificar millones de páginas web desde que fue introducido en 1996.

La idea detrás del algoritmo de PageRank es que una página web es más importante si es enlazada por otras páginas importantes. El algoritmo asigna una puntuación de importancia a cada página web, denominada "rank", y esta puntuación se utiliza para determinar el orden en el que las páginas aparecen en los resultados de búsqueda.

El algoritmo de PageRank opera asignando una calificación inicial, conocida como “PageRank inicial”, a cada página web. Posteriormente, utiliza esta calificación para determinar el PageRank de cada página, basándose en los enlaces que recibe de otras páginas. El algoritmo sigue un proceso repetitivo para calcular el PageRank final de cada página web.

Uno de los aspectos clave del algoritmo de PageRank es el concepto de "voto" o enlace. Cada enlace que una página recibe de otra página se considera como un voto de confianza en la importancia de la página enlazada. El algoritmo asigna un peso diferente a cada enlace en función de la importancia de la página que proporciona el enlace. Por ejemplo, un enlace de un sitio web altamente valorado tendrá más peso que un enlace de un sitio web menos valorado.

El algoritmo también tiene en cuenta el "efecto de surtido" en el que una página con muchos enlaces puede transferir más "votos" o importancia que una página con pocos enlaces. Por lo tanto, una página con un gran número de enlaces de sitios web importantes puede tener un PageRank más alto que una página con solo unos pocos enlaces de sitios web menos importantes.

Además, el algoritmo de PageRank también tiene en cuenta el concepto de "damping factor", que tiene en cuenta la posibilidad de que los usuarios no sigan enlaces al azar y terminen en una página no relacionada con su búsqueda. Esto se logra reduciendo la transferencia de importancia entre página y página.

2.2.1 Principales usos y características de PageRank

PageRank se basa en la idea de que una página es importante si es enlazada por otras páginas importantes. El algoritmo asigna un valor de "rango" a cada página, que indica su importancia. Las páginas con un valor de rango más alto son consideradas más importantes que las páginas con un valor de rango más bajo.

La implementación original de PageRank utilizaba una matriz de enlaces para representar las relaciones entre las páginas. Cada fila de la matriz representa una página (o nodo), y cada columna representa una página enlazada. El valor en cada celda de la matriz indica el número de enlaces desde la página de la fila a la página de la columna. Esta matriz se utilizó para calcular un vector de rango que representaba la importancia de cada página.

El algoritmo PageRank es fundamental para el funcionamiento de Google, ya que se utiliza para determinar el orden en que aparecen los resultados de búsqueda. Sin embargo, también ha sido utilizado en otras aplicaciones, como el análisis de redes sociales, la recomendación de contenido y la clasificación de noticias.

En el ámbito de las redes sociales, el algoritmo de PageRank se utiliza para identificar los usuarios más influyentes en una red. Esto se logra mediante la medición de la cantidad y la calidad de los enlaces entrantes a un perfil. A los usuarios con un mayor número de enlaces entrantes de usuarios influyentes se les considera también más influyentes.

En la recomendación de contenido, el algoritmo de PageRank se utiliza para sugerir contenido relacionado a los usuarios basándose en las páginas que han visitado previamente. El algoritmo asume que, si un usuario ha visitado una página, es probable que esté interesado en el contenido relacionado.

Finalmente, el algoritmo de PageRank también ha sido utilizado en la clasificación de noticias. En este caso, se utiliza para determinar la importancia de una noticia en relación con otras noticias. Esto se logra mediante la medición de la cantidad y la calidad de los enlaces entrantes entre los principales portales de noticias.

2.3 Representación gráfica del algoritmo de PageRank

La representación gráfica de PageRank se conoce como un grafo dirigido, donde cada nodo representa una página web y cada arco representa un enlace entre páginas. En este grafo, la importancia de cada página se representa mediante un valor de rango asignado al nodo correspondiente.

El algoritmo PageRank asigna una importancia inicial a cada página y luego a través de iteraciones va actualizando el rango de importancia en función de los enlaces entrantes y salientes de cada página. El proceso finaliza cuando se alcanza un estado estable de rango de importancia de las páginas.

Por lo general se representa gráficamente los nodos más importantes con un tamaño más grande y los menos importantes con un tamaño más pequeño, además las flechas entrantes tienen un peso mayor que las salientes.

Es importante destacar que el algoritmo de PageRank también tiene en cuenta el factor denominado como "damping factor" o factor de amortiguamiento, que permite evitar la perpetuación de los ciclos de enlaces, esto es que ciertas páginas tienen un cierto porcentaje de posibilidades de ser seleccionadas aleatoriamente, lo que permite tener un mejor rendimiento y precisión en la asignación del rango de importancia a las páginas.

De la patente de Google del algoritmo de PageRank podemos recuperar la representación gráfica, que se ve de la siguiente manera, *(US9165040B1 - Producing a Ranking for Pages Using Distances in a Web-Link Graph - Google Patents, 2006) [5].*

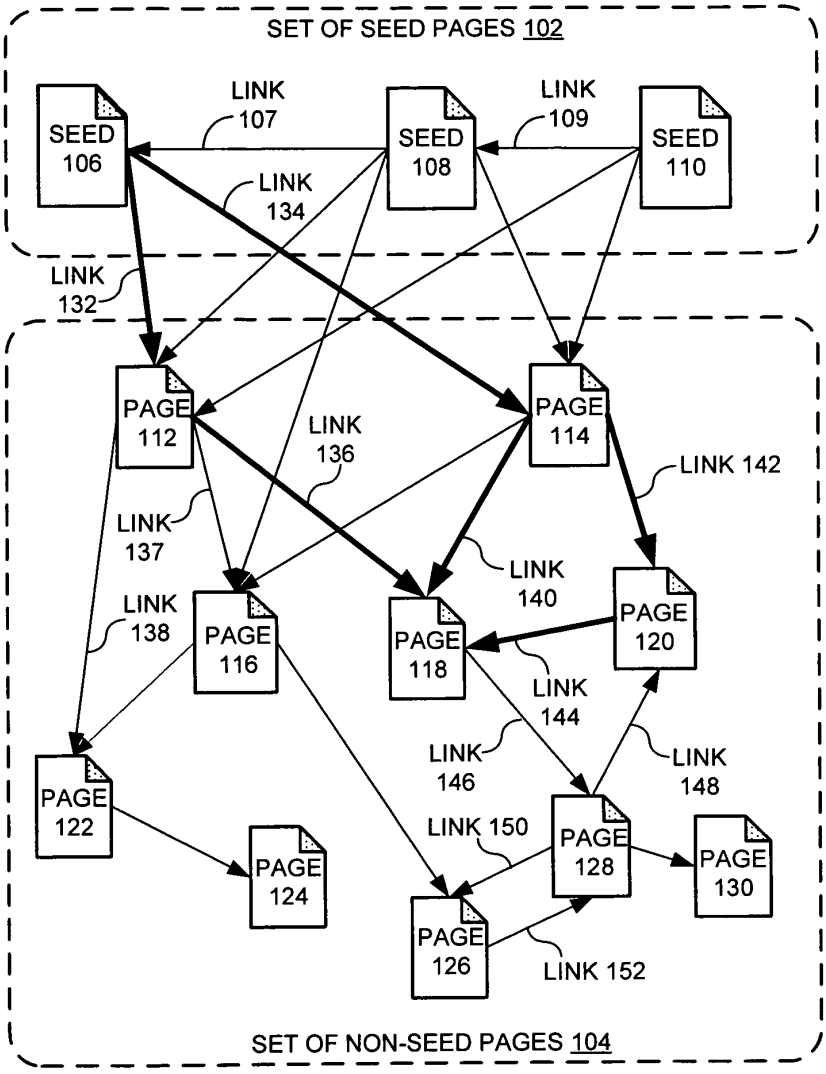

Ilustración 1 Representación gráfica de PageRank

La imagen que se presenta es una representación gráfica del algoritmo PageRank, una técnica fundamental en el ámbito de la optimización de motores de búsqueda y la evaluación de la relevancia de las páginas web. En esta visualización, se puede observar un conjunto de nodos interconectados, que simbolizan las páginas web, y las flechas que los unen representan los enlaces hipertextuales entre estas páginas. Algunos nodos parecen más grandes y centralizados, mientras que otros son más pequeños y periféricos. Estas diferencias en el tamaño y la ubicación de los nodos indican la importancia y la influencia de cada página en la red. Los nodos más grandes reflejan las páginas con un mayor PageRank, lo que significa que están vinculados a través de enlaces desde otras páginas importantes. En contraste, los nodos más pequeños pueden ser páginas menos influyentes o menos conectadas en la red. En resumen, esta representación gráfica del algoritmo PageRank ilustra de manera visual cómo se establecen y distribuyen las relaciones de importancia entre las páginas web en función de sus conexiones y la estructura de enlaces.

2.4 Stop-words

Las stop words son palabras comunes y de poco significado que se suelen filtrar de un texto antes de realizar alguna tarea de procesamiento de lenguaje natural. Estas palabras incluyen artículos, preposiciones, conjunciones y pronombres, y se consideran stop words porque no aportan mucha información semántica al texto y pueden ser descartadas sin afectar su significado general.

Algunos ejemplos de stop words en el idioma inglés incluyen "the", "a", "an", "and", "or", "of", "in", "to", "with", "for", "on", "by", etc. Cada idioma puede tener una lista diferente de stop words, y esta lista puede ser personalizada según las necesidades específicas de una tarea.

Eliminar las stop words antes de realizar un análisis de texto puede mejorar la eficiencia y la precisión del análisis, ya que permite enfocarse en las palabras clave y relevantes en el texto. Muchas bibliotecas de procesamiento de lenguaje natural, como NLTK en Python, tienen listas incorporadas de stop words que se pueden utilizar para filtrar estas palabras en un análisis de texto. Sin embargo, es importante tener en cuenta que la definición de stop words puede variar en función del contexto y la aplicación, por lo que es posible que deba crear su propia lista de stop words en función de sus necesidades específicas [13].

La clasificación de textos es un área de investigación activa en la recuperación de información y el procesamiento del lenguaje natural. Una herramienta fundamental en la clasificación de textos es una lista de palabras 'no válidas' (lista de palabras no válidas) que se utiliza para identificar palabras frecuentes que es poco probable que ayuden en la clasificación y, por lo tanto, se eliminan durante el preprocesamiento. (Hao, 2008).

2.5 Tokenize

La tokenización representa un proceso vital dentro del campo del procesamiento del lenguaje natural, el cual consiste en desglosar una cadena de texto en fragmentos más pequeños denominados "tokens". Estos tokens representan unidades lógicas que encapsulan partes significativas del texto, tales como palabras o frases.

Este procedimiento adquiere una importancia crucial en el tratamiento del texto, dado que posibilita una manipulación más eficiente y precisa del mismo. Al segmentar el texto en tokens, se abre la puerta a un análisis minucioso de cada elemento, lo que facilita una comprensión más profunda del contenido textual y la extracción de información relevante.

La tokenización puede llevarse a cabo de diversas maneras, como la segmentación por palabras, frases o párrafos, y la elección del método adecuado puede variar según las aplicaciones y los contextos específicos.

En el ámbito de la programación en Python, disponemos de varias bibliotecas de procesamiento del lenguaje natural, tales como NLTK y SpaCy, que ofrecen funciones integradas de tokenización, simplificando así el proceso de desglose del texto en tokens de manera eficiente. Asimismo, en caso de requerir un enfoque más personalizado, es posible implementar una función de tokenización propia, adaptada a las necesidades particulares del análisis textual en cuestión.

2.6 Expresiones regulares

Las expresiones regulares son una forma de describir patrones de texto mediante una sintaxis especial. Son una herramienta muy poderosa para manipular y procesar texto, y se utilizan en muchas aplicaciones, como la búsqueda y reemplazo de texto, la validación de direcciones de correo electrónico y números de teléfono, y la extracción de información de texto no estructurado.

Las expresiones regulares se basan en un conjunto de reglas que definen cómo los caracteres deben combinarse para formar un patrón. Por ejemplo, una expresión regular podría definir un patrón para una dirección de correo electrónico, que incluiría caracteres alfanuméricos seguidos de un símbolo de arroba y terminando con un nombre de dominio.

En Python, las expresiones regulares se pueden manipular mediante la biblioteca "re". Esta biblioteca proporciona funciones que permiten compilar y usar expresiones regulares para buscar y manipular texto.

Es importante tener en cuenta que el aprendizaje de las expresiones regulares requiere una cierta habilidad en programación y una comprensión sólida de la sintaxis y las reglas de las expresiones regulares. Sin embargo, una vez que se comprende su funcionamiento, las expresiones regulares pueden ser una herramienta muy poderosa para procesar y manipular texto en muchas aplicaciones diferentes.

Capítulo 3 | Herramientas y aplicaciones

3.1 Python

El lenguaje de programación Python es uno de los lenguajes de programación más populares y versátiles del mercado actual. Desarrollado por Guido van Rossum en 1991, Python se ha convertido en uno de los lenguajes de programación más utilizados y accesibles tanto para principiantes como para programadores experimentados. En este artículo exploraremos las principales características de Python, su uso en diferentes aplicaciones, sus ventajas y desventajas y su lugar en el mercado de los lenguajes de programación.

Una de las características clave de Python es su sintaxis legible de alto nivel. A diferencia de otros lenguajes de programación, Python se escribe y se lee como un lenguaje humano, lo que facilita que los principiantes lo entiendan y aprendan. Además, Python tiene una amplia gama de bibliotecas y herramientas que permiten a los programadores resolver problemas complejos de manera más eficiente y rápida.

Python se usa ampliamente en diferentes dominios y aplicaciones, incluido el análisis de datos, el aprendizaje automático, las redes y la robótica. Por ejemplo, en el análisis de datos, Python es una de las herramientas más populares y versátiles debido a su capacidad para manipular y visualizar de manera eficiente grandes cantidades de datos. En el aprendizaje automático, Python también es una herramienta valiosa debido a su extensa biblioteca de aprendizaje automático y soporte para diferentes algoritmos de aprendizaje automático.

Otra ventaja importante de Python es su facilidad de aprendizaje. A diferencia de otros lenguajes de programación más complejos, Python es accesible para los programadores principiantes y puede ser aprendido en un corto período de tiempo. Además, la comunidad de desarrolladores de Python es muy activa y existen muchos recursos disponibles, incluyendo documentación, tutoriales y comunidades en línea, para ayudar a los programadores a aprender y resolver problemas.

Python es un lenguaje de programación poderoso, versátil y fácil de aprender. Con él, puedes abordar una amplia variedad de proyectos, desde la automatización de tareas hasta el desarrollo de aplicaciones web y móviles. *González Duque, R. (Fecha desconocida). Python para todos. Editorial desconocida [4].*

3.2 Python y sus librerías

Las bibliotecas de Python son colecciones de módulos preconstruidos que proporcionan funciones y herramientas adicionales para la programación en Python. Estas bibliotecas permiten a los desarrolladores resolver problemas comunes y complejos más fácilmente y rápidamente, sin tener que escribir todo el código desde cero.

Algunas de las bibliotecas más populares en Python incluyen:

- NumPy: una biblioteca para el cálculo numérico y la manipulación de arrays multidimensionales [6].
- Pandas: una biblioteca para el análisis de datos y la manipulación de tablas de dato [7].
- Matplotlib: una biblioteca para la creación de gráficos y visualizaciones de datos [8].
- Scikit-learn: una biblioteca para el aprendizaje automático y el análisis de datos [9].
- Tensorlow y PyTorch: bibliotecas para el aprendizaje profundo y la inteligencia artificial [10].

Estas son solo algunas de las muchas bibliotecas disponibles en Python, y nuevas bibliotecas se agregan constantemente a la comunidad de desarrolladores de Python. Estas bibliotecas son una parte esencial de la riqueza y la versatilidad de Python, y pueden ser una gran ayuda para los desarrolladores en la resolución de problemas complejos y en la construcción de aplicaciones poderosas.

3.3. Procesamiento del lenguaje natural

El procesamiento del lenguaje natural (NLP, por sus siglas en inglés) es una disciplina de la inteligencia artificial que se enfoca en permitir que las computadoras entiendan y procesen el lenguaje humano de una manera más natural. Con la creciente importancia de la tecnología en la sociedad actual, la NLP se ha convertido en un campo de investigación y desarrollo muy activo y prometedor.

3.3.1 Historia

El NLP tiene sus raíces en la década de 1950, cuando los científicos comenzaron a investigar la forma en que las computadoras podrían procesar y comprender el lenguaje humano. Durante las siguientes décadas, se desarrollaron diversos enfoques y técnicas para abordar este desafiante problema, incluyendo la gramática formal, la representación lógica y el aprendizaje automático. En los últimos años, con el avance de la tecnología y la disponibilidad de grandes cantidades de datos, el NLP ha experimentado un auge sin precedentes y se ha convertido en una disciplina fundamental en la investigación en inteligencia artificial.

3.3.2 Aplicaciones

NLP tiene una amplia gama de aplicaciones en la industria y la sociedad, incluidos chatbots, traducción automática, análisis de sentimientos, generación de texto y más. Por ejemplo, los chatbots basados en NLP se pueden usar para responder preguntas y brindar información de manera rápida y eficiente, mientras que las aplicaciones de traducción automática pueden ayudar a superar las barreras del idioma y mejorar la comunicación entre personas de diferentes países. Además, NLP se puede utilizar para analizar grandes cantidades de datos textuales y extraer información valiosa, como tendencias y patrones, que de otro modo sería difícil de detectar.

3.3.3 Desafíos

A pesar de los logros notables en el NLP, aún hay desafíos significativos que deben ser abordados. Uno de los mayores desafíos es la ambigüedad en el lenguaje humano. Por ejemplo, las mismas palabras pueden tener diferentes significados en diferentes contextos, lo que puede ser confuso para las computadoras. Otro desafío importante es la comprensión del lenguaje en contexto, lo que requiere una comprensión profunda de la estructura del lenguaje y la relación entre las palabras y las frases. Además, el NLP también se enfrenta a desafíos en la gestión de los datos, incluyendo la corrección de errores, la eliminación de ruido y la normalización de los datos de entrada.

En resumen, el procesamiento del lenguaje natural es una subdisciplina de la inteligencia artificial que se centra en la comprensión y manipulación del lenguaje humano por parte de las computadoras. El campo ha crecido rápidamente en los últimos años debido a la disponibilidad de grandes cantidades de datos y la mayor capacidad de las computadoras para procesar y analizar información. Sin embargo, todavía existen muchos desafíos en la PNL, como la ambigüedad en el lenguaje

humano y la comprensión del lenguaje en contexto. Además, NLP tiene una amplia gama de aplicaciones prácticas, desde minería de texto y clasificación de documentos hasta resúmenes automáticos y traducción automática. En resumen, la PNL es un campo en crecimiento y se espera que siga siendo una parte importante de la investigación en inteligencia artificial y tecnología de la información en el futuro.

Capítulo 4 | Conceptualización del algoritmo

Bajo la línea de información que se sigue en el presente trabajo, la implementación y conceptualización de nuestro algoritmo, es importante, el siguiente Seudocódigo detalla la estructura que sigue el código:

1. Abrimos el documento de txt y lo guardamos en una variable
2. Importamos las librerías requeridas.
3. Las cadenas a continuación son las consultas que realizaremos
4. Removemos las etiquetas de respuesta (A), (B), (C), (D) del texto.
5. Obtenemos las palabras comunes en el idioma inglés que no aportan significado en el texto (stopwords).
6. Separamos el texto en palabras individuales (tokenización). (Tokenizar significa separar una cadena de texto en piezas más pequeñas llamadas tokens, que pueden ser palabras individuales, signos de puntuación, etc.).
7. Preprocesamos el corpus eliminando palabras vacías (stop words) y signos de puntuación.
8. Realizamos la consulta del texto eliminando palabras vacías y signos de puntuación.
9. Obtenemos los índices donde se encuentra cada palabra de la consulta en el corpus.
10. Graficamos la frecuencia de las palabras en el corpus preprocesado.
11. Guardamos el corpus preprocesado.
12. Generamos una nube de palabras a partir del corpus preprocesado.

Representación Gráfica del algoritmo

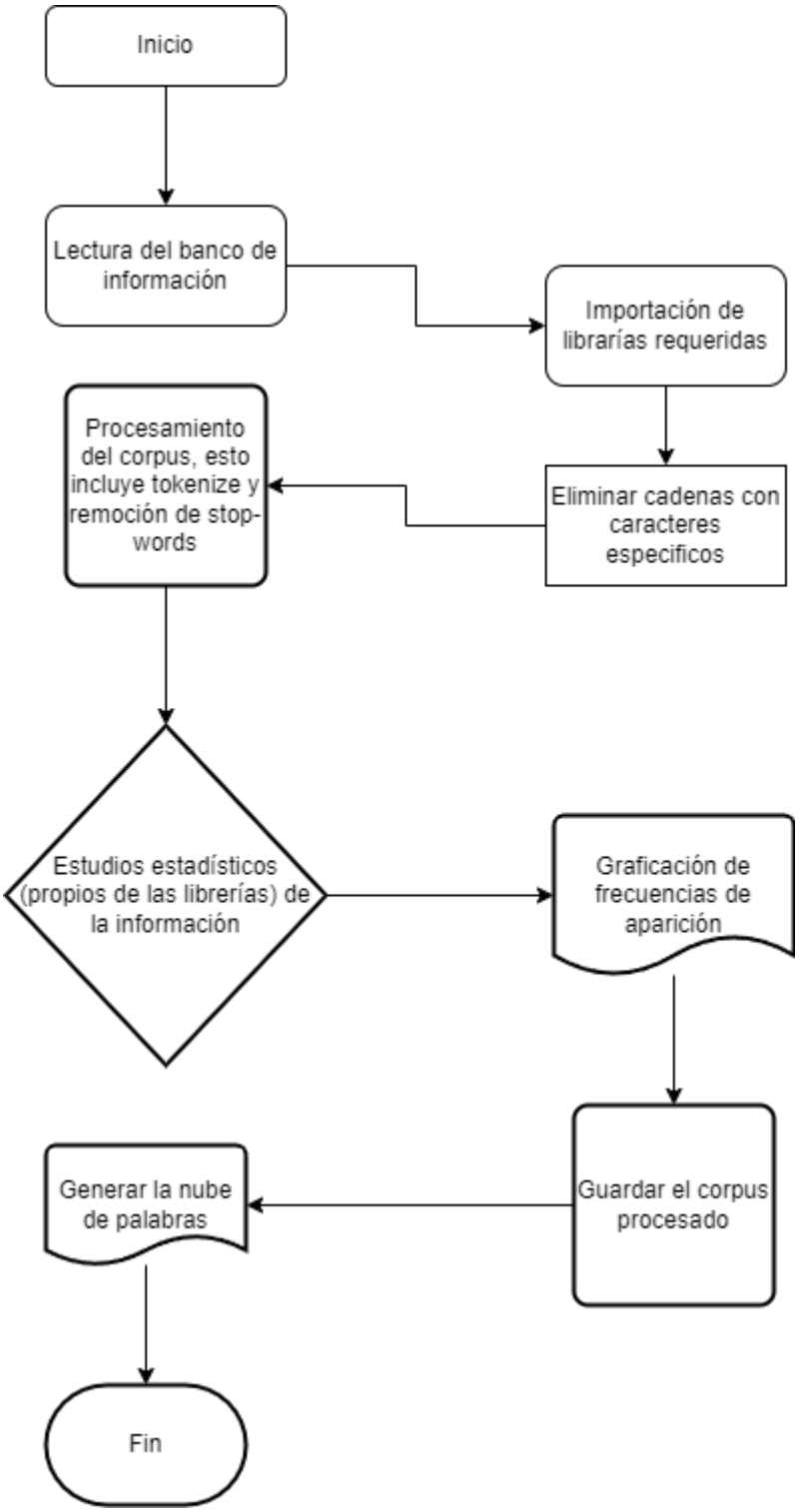

Fig. 1 Diagrama de flujo propuesto

Consideraciones

Es importante hacer algunas consideraciones, que pueden ser importantes, la principal de ellas es el uso de archivos de txt en vez de formatos más desarrollados como PDF o DOCX, debido a perdida de información en el traslado de información de un archivo a nuestro código, o de políticas de seguridad y privacidad que no hacen posible la extracción de información de un documento determinado, es por ello que al eliminar estos factores, se obtiene un resultado más preciso al momento de ejecutar el código.

Capítulo 5 | Implementación y primeras versiones

5.1 Primera versión de código en lenguaje Python

```
# -*- coding: utf-8 -*-
"""
Created on Mon Aug 30 20:16:05 2021

@author: Aaron Ramirez
"""
#Abrimos el documento de txt y lo guardamos en una variable
with open('TOEFL Reading Comprehension 1.txt','r',encoding= "utf8") as
miarchivo:
    texto = miarchivo.read()

#Importamos las librerias requeridas.
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import string
import nltk
import pandas as pd
import matplotlib.pyplot as plt
from wordcloud import WordCloud

from collections import Counter
from collections import OrderedDict


#las cadenas a continuacion son las consultas que realizaremos
#cadena='separation anxiety in infancy (i.e. up to two years of age) and in
preschool children, particularly separation of a child from its mother'
#cadena='the toxicity of organic selenium compounds'
cadena='language development in infancy and pre-school age'
#cadena= '!#$%&()*+,-./:;<=>?@[\]^_`{|}~'
#cadena='of not few so on where as no how d before shouldve has weren than
will '
#cadena="Perro gato"
texto = texto.replace('(A)', '')
texto = texto.replace('(B)', '')
texto = texto.replace('(C)', '')
texto = texto.replace('(D)', '')

#obtenemos las stop_words en el mismo lenguaje que el corpus
stop_words= set(stopwords.words('english'))

word_tokens = word_tokenize(texto) #tookenizar significa utilizar toda la
palabra y no solo un caracter
```

```
word_tokens1 = word_tokenize(cadena)
##### PREPROCESAMIENTO DEL TEXTO #####
word_tokens = list(filter(lambda token : token not in
string.punctuation,word_tokens,)) #Eliminamos caracteres de puntuacón del
corpus
word_tokens1= list(filter(lambda token : token not in
string.punctuation,word_tokens1)) #Eliminamos caracteres de puntuación de la
consulta
filtro=[] #Declaramos una variable de tipo lista que contendrá el corpus una
vez finalizado el preprocesamiento
filtro1=[]  #Declaramos una variable de tipo lista que contendrá la consulta
una vez finalizado el preprocesamiento
aux=[]#Utilizamos una variable auxiliar para realizar la consulta
for palabra in word_tokens: #iniciamos el ciclo para eliminar stop words
    if palabra not in stop_words:
        filtro.append(palabra)

for i in word_tokens1:
    if i not in stop_words:
        filtro1.append(i)

c=Counter(filtro) # Obtenemos la propiedad contador de la libreria usada
fdist=nltk.FreqDist(filtro) # Usamos una funcion de la libreria para obtener
la frecuencia el
                           #la distribución del corpus preprocesado
fdist.plot(20,cumulative=True) #Graficamos los primeros 20 téminos más
usuales
##### GUARDAMOS EL CORPUES PREPOSESADO #####
y=OrderedDict(c.most_common())
with open('salida.txt','w',encoding='utf-8') as file:
    for k,v in y.items():
        file.write(f'{k} ' )

filename = "salida.txt"
with open(filename,encoding='utf-8') as f:
 mytext = f.read()

wordcloud = WordCloud().generate(mytext)
#%pylab inline

plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.savefig("resultado.png")
plt.show()
```

Código 1 Versión completa del código

5.2 Desarrollo del código

A continuación, el algoritmo utilizado de manera más detallada.

5.2.1 Variables

En primera instancia, y como variable central y principal, tenemos nuestro corpus, sin procesar, y sin ningún procesamiento.

```
with open('TOEFL Reading Comprehension 1.txt','r',encoding= "utf8") as
miarchivo:
  texto = miarchivo.read()
```

Código 2 Declaración de variables

5.2.2 Librerías

Posterior a ello, importamos las librerías principales que usaremos.

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import string
import nltk
import pandas as pd
import matplotlib.pyplot as plt
from wordcloud import WordCloud

from collections import Counter
from collections import OrderedDict
```

Código 3 Librerías usadas

5.2.3 Cadenas

El código se basa en un programa de recuperación y aun se utiliza la versión de búsqueda de una cadena dentro del corpus.

```
#las cadenas a continuacion son las consultas que realizaremos
#cadena='separation anxiety in infancy (i.e. up to two years of age) and in
preschool children, particularly separation of a child from its mother'
#cadena='the toxicity of organic selenium compounds'
cadena='language development in infancy and pre-school age'
#cadena= '!#$%&()*+,-./:;<=>?@[\]^_`{|}~'
#cadena='of not few so on where as no how d before shouldve has weren than
will '
#cadena="Perro gato"
```

Código Consideraciones generales

5.2.4 Aspectos a considerar

La siguiente parte de nuestro código, es cambiar las cadenas “(A), (B),(C),(D)”, en primera instancia porque no entran dentro de los signos de puntuación, también es importante saber el contexto, pues al tratarse de un corpus de ensayo para TOEFL, existen preguntas, y por ende opciones de respuestas, es por ello que es importante eliminar estas pastes, para que no afecten los resultados.

```
texto = texto.replace('(A)', '')
texto = texto.replace('(B)', '')
texto = texto.replace('(C)', '')
texto = texto.replace('(D)', '')
```
Código 4 Consideraciones de cadenas relevantes

5.2.5 Stop-Words

Las stop-words son imprescindibles dentro del procesamiento de texto, es por ello que nuestro siguiente paso es obtenerlas como un punto importante dentro de la ejecución del código.

```
#obtenemos las stop_words en el mismo lenguaje que el corpus
stop_words= set(stopwords.words('english'))
```
Código 5 Stop Words

5.2.6 Tokenize

Una vez hecho eso, la siguiente parte consiste en aplicar el *tokenize* de nuestra información, en primera instancia de nuestro corpus principal, como también como con nuestras cadenas de búsqueda.

```
word_tokens = word_tokenize(texto) #tookenizar significa utilizar toda la
palabra y no solo un caracter
word_tokens1 = word_tokenize(cadena)
```
Código Tokenización

5.2.7 Procesamiento del texto

Comenzamos al procesamiento del texto, eliminamos, stop-words, signos de puntuación, y como anteriormente ya hemos eliminado algunos factores importantes obtenemos un texto de salida que puede observarse en el apartado de figuras y tablas; Fig. 1 si imprimimos la variable filtro.

```
Word_tokens = list(filter(lambda token : token not in
string.punctuation,word_tokens,)) #Eliminamos caracteres de puntuacón del
corpus
word_tokens1= list(filter(lambda token : token not in
string.punctuation,word_tokens1)) #Eliminamos caracteres de puntuación de la
consulta
filtro=[] #Declaramos una variable de tipo lista que contendrá el corpus una
vez finalizado el preprocesamiento
filtro1=[]  #Declaramos una variable de tipo lista que contendrá la consulta
una vez finalizado el preprocesamiento
aux=[]#Utilizamos una variable auxiliar para realizar la consulta
for palabra in word_tokens: #iniciamos el ciclo para eliminar stop words
    if palabra not in stop_words:
        filtro.append(palabra)
```

```
for I in word_tokens1:
    if I not in stop_words:
```

Código 6 Procesamiento de la información `filtro1.append(i)`

.

5.2.8 Manejo de resultados

Obtennos los resultados y C es una variable que nos dice cuántas veces se repite una palabra en nuestro corpus Fig.2, que nos servirá más adelante y usamos la función de obtención de distribución de frecuencia de nuestro corpus.

```
c=Counter(filtro) # Obtenemos la propiedad contador de la libreria usada
fdist=nltk.FreqDist(filtro) # Usamos una funcion de la libreria para obtener
la frecuencia el
```

Código 7 Manejo de resultados

5.2.9 Salida

A continuación, se guardan los resultados que se obtienen y se guarda en un archivo de texto, la salida de nuestro código será nuestro corpus ya procesado.

```
y=OrderedDict(c.most_common())
with open('salida.txt','w',encoding='utf-8') as file:
    for k,v in y.items():
        file.write(f'{k} ' )

filename = "salida.txt"
```

Código 8 Salidas

5.2.10 Resultados gráficos

Una vez realizada la salida de nuestro texto, abriremos nuestro corpus, y lo guardamos en una variable que permita el manejo de información.

```
filename = "salida.txt"
with open(filename,encoding='utf-8') as f:
 mytext = f.read()
```

Código 9 Salidas graficas

5.2.11 Nube de palabras

Los resultados obtenidos se muestran de dos maneras, en la Fig. 3 observamos nuestra nube de palabras, y en la Fig. 4 se muestra nuestra grafica de términos más comunes del texto, ambas bajo la representación gráfica.

```
wordcloud = WordCloud().generate(mytext)
#%pylab inline

plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.savefig("resultado.png")
```

```
plt.show()
```

Código 10 Nube de palabras

Figuras

Figura 1

Fig. 2 Variable Filtro

La representación gráfica proporcionada en la figura anterior ofrece una instantánea del output obtenido del proceso. Dado el tamaño considerable del corpus completo, se ha optado por exhibir una muestra considerable pero manejable para fines de visualización. Esta muestra procesada ilustra de manera efectiva la eliminación exitosa de las palabras de parada(stop-words), las cuales son términos comunes que generalmente se excluyen en el análisis de texto debido a su limitado valor semántico. Al comparar este resultado con el corpus original, se puede observar cómo estas palabras de parada(stop-words), que incluyen pronombres, preposiciones y conjunciones, han sido correctamente filtradas. Este proceso es de vital importancia, ya que, al remover el ruido lingüístico innecesario, el enfoque analítico se dirige hacia las palabras sustantivas y significativas, lo que potencia la calidad y la profundidad del análisis textual realizado.

Figura 2

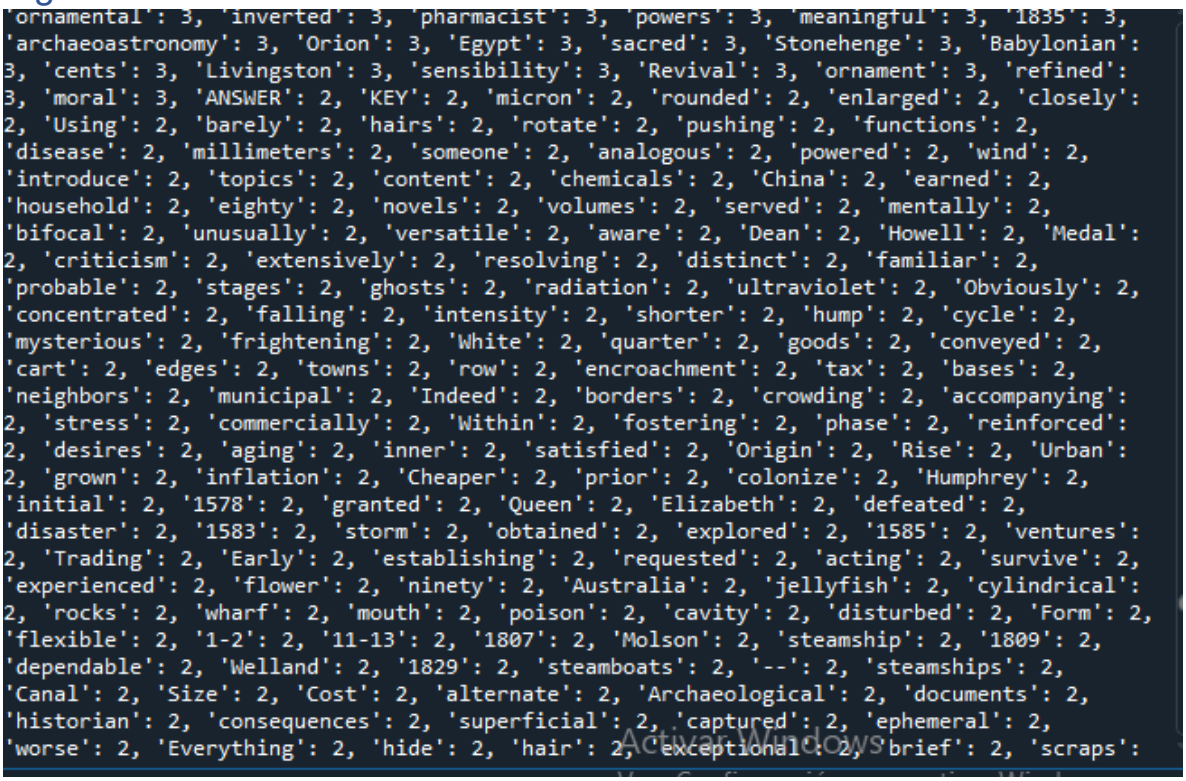

Fig. 3 Frecuencia de aparición de términos en el corpus.

La figura precedente exhibe el desenlace derivado de la enumeración de las palabras, un procedimiento llevado a cabo durante el procesamiento textual. Esta información resulta de gran significancia al momento de identificar las palabras que ocurren con mayor frecuencia en el corpus. Dicha frecuencia de aparición posee un papel esencial al permitirnos discernir las palabras de mayor relevancia dentro del conjunto. Al efectuar este análisis cuantitativo, se desvela una perspectiva valiosa sobre qué términos ejercen una influencia más pronunciada en el corpus en cuestión.

Figura 3

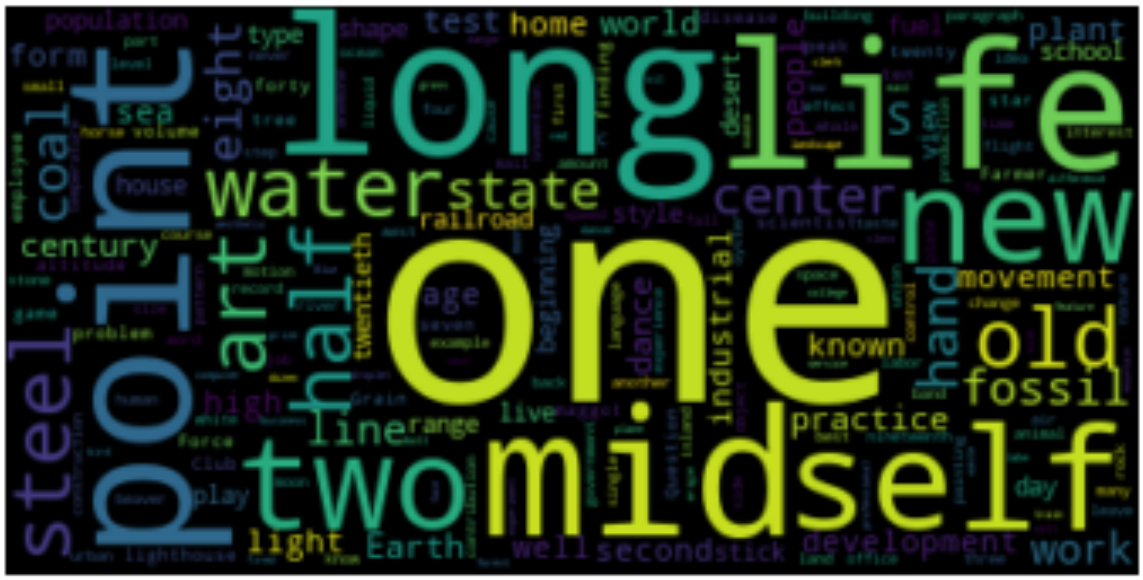

Fig. 4 Nube de palabras

La ilustración previa presenta de manera más visual el desenlace del recuento de palabras según su incidencia. En esta representación gráfica, se refleja que a medida que la incidencia de una palabra aumenta, su posición en la jerarquía también se eleva. Como consecuencia directa, esta representación proporciona una perspectiva más efectiva para visualizar el corpus que ha sido sometido a procesamiento.

Figura 4

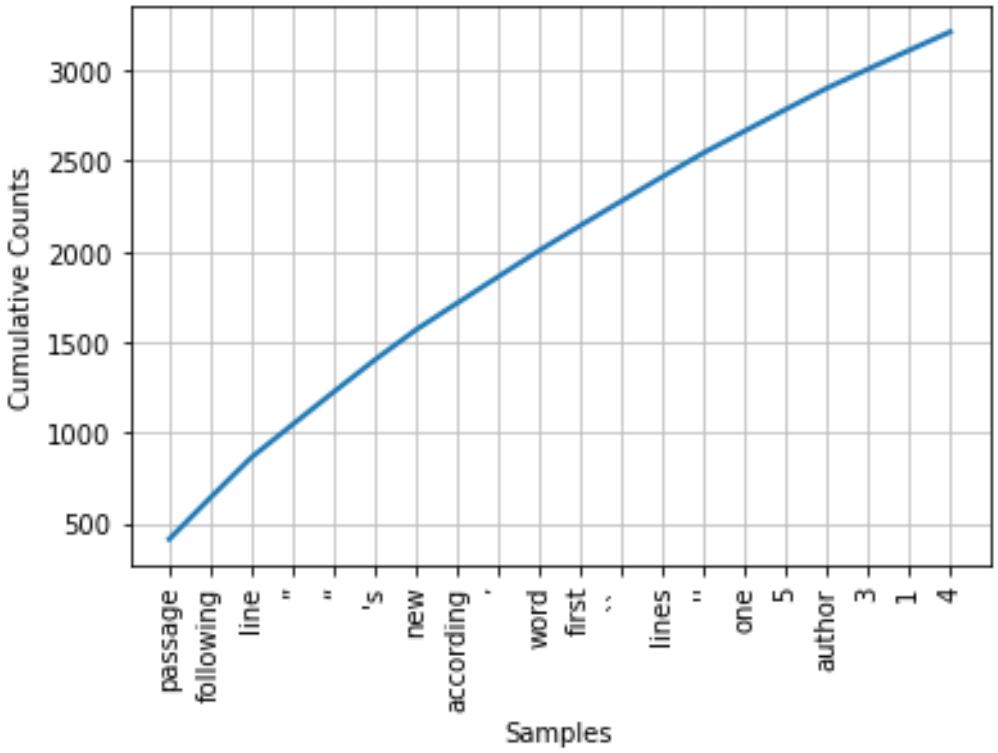

Fig. 5 La figura previamente expuesta exhibe un conjunto de las 20 palabras de mayor relevancia obtenidas. Siguiendo esta misma premisa y al considerar los resultados derivados del análisis del corpus, estas palabras han sido destacadas por su importancia intrínseca en el contexto del estudio.

Capítulo 6 | Conclusiones y trabajo a futuro

Conclusión

El análisis de textos es una tarea importante en diferentes áreas, desde la investigación científica hasta la toma de decisiones en empresas. En este sentido, el código presentado utiliza una serie de herramientas para analizar un corpus de texto y responder a consultas específicas.

El trabajo comienza listando conceptos importantes para entender la conceptualización e implementación del código, para poder explicar de mejor manera nuestros resultados, y por ello es importante entender cómo se estructura la información, y como procesarla.

Una de las tareas importantes en el preprocesamiento del texto es la eliminación de stop words, es decir, palabras que no aportan información relevante para el análisis. Para ello, se utilizan herramientas del procesamiento del lenguaje natural, para obtener una lista de stop words en inglés.

La información y los resultados que se muestran anteriormente nos permiten observar los puntos importantes del proceso de recuperación de la información, han demostrado que es posible manejar volúmenes de información considerable, también permite ver de mejor manera hacia donde se puede dirigir de mejor manera la curva de aprendizaje de la formación que se quiere obtener.

En conclusión, el presente trabajo muestra cómo se pueden utilizar diferentes herramientas para el análisis de textos, desde la eliminación de stop words hasta la visualización de resultados. La utilización de estas herramientas puede resultar útil en diferentes áreas, como la investigación científica, la gestión empresarial o el análisis de redes sociales, entre varios otros usos que se le pueden dar a esta información.

Trabajos a futuro

El Presente trabajo, ha demostrado el uso de diferentes herramientas que permiten el manejo de información, y procesamiento, el trabajo a futuro puede tomar el presente código y perfeccionar para el mejor empleo de resultados, algunos aspectos importantes que se sugieren a tener en cuenta para una mejora significativa del presente proyecto es tener en cuenta los siguientes conceptos:

- Los dígitos, números y caracteres numéricos en la recuperación de información dependiendo del contexto al que se aplica.
- Uso de expresiones regulares y su importancia en el manejo de información.
- Las contracciones del lenguaje, en distintos lenguajes existen contracciones, que pueden variar el resultado de nuestro procesamiento de la información.
- Palabras validas, que palabras en determinados lenguajes son consideradas validas forman una ‘palabra y no un cumulo de caracteres.

Bibliografía

[1] Soto, D. M. N., & Tzompantzi, J. P. M. (Año no proporcionado). EL USO DE APPS PARA EL APRENDIZAJE DEL INGLÉS A TRAVÉS DEL MODELO B-LEARNING.

[2] Gallardo Arreola, A., & Gallardo Arreola, A. N. A. L. I. (2022). Propuesta de un taller de estrategias para mejorar la comprensión auditiva del inglés en alumnos de lengua extranjera FGU–BUAP a partir de un estudio cuasiexperimental (Tesis de maestría, Benemérita Universidad Autónoma de Puebla).

[3] Neve Brito, M. G. (2017). “Moodle” como herramienta para el aprendizaje de inglés como lengua extranjera: un estudio de caso.

Abadal, E., & Codina, L. (2005). Recuperación de información. Bases de Datos Documentales: Características, funciones y método, 29-92.

Fernandez, A. (2013). Python 3 al descubierto. Alfaomega Grupo Editor.

[4] González Duque, R. (2011). Python para todos.

[5 US9165040B1 - Producing a ranking for pages using distances in a web-link graph – Google Patents. (2006, October-12). Google.com. https://patents.google.com/patent/US9165040B1/en?scholar

[6] NumPy. (2023). Recuperado el 4 de febrero de 2023, de https://numpy.org/

[7] Pandas - Python Data Analysis Library. (2023). Recuperado el 4 de febrero de 2023, de https://pandas.pydata.org/

[8] Matplotlib — Visualization with Python. (2022). Recuperado el 4 de febrero de 2023, de https://matplotlib.org/

[9] Scikit-learn: machine learning in Python — scikit-learn 1.2.1 documentation. (2023). Recuperado el 4 de febrero de 2023, de https://scikit-learn.org/stable/

[10] TensorFlow. (2023). Recuperado el 4 de febrero de 2023, de https://www.tensorflow.org/?hl=es-419

[11] Hao, L., & Hao, L. (2008, Diciembre). Automatic identification of stop words in Chinese text classification. En 2008 International Conference on Computer Science and Software Engineering (Vol. 1, pp. 718-722). IEEE.

[12] Méndez Rodríguez, E. M. (2001). Metadatos y recuperación de información: estándares, problemas y aplicabilidad en bibliotecas digitales.

[13] Cuba Rodríguez, Y., & Olivera Batista, D. (2018). Los metadatos, la búsqueda yrecuperación de información desde las Ciencias de la Información. E-Ciencias de la Información, 8(2), 146-158.