Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Estado de México

TC3002B

Desarrollo de Aplicaciones avanzadas de ciencias Computacionales

Escuela de ingeniería y ciencias

Grupo 301

Evidencia 1

Funcionalidad y Documentación de Detección de plagio y ciencias computacionales

Eduardo Acosta Hernández	A01375206
Renata Montserrat De Luna Flores	A01750484
Roberto Valdez Jasso	A01746863

Profesores

Raúl Monroy

Jorge Adolfo Ramírez Uresti

Ariel Ortiz Ramírez

Miguel González Mendoza

Fecha de entrega

26 de mayo de 2023

Tabla de Contenido

1. Resumen

La detección de plagio y reutilización de texto es de suma importancia en la actualidad, ya que caer en esto es cometer un delito. Es por esto que se han desarrollado diversas técnicas para detectarlo. En este documento, desarrollaremos y describiremos la implementación del software desarrollado para lograr la detección tanto de plagio como de reutilización de texto.
Para lograr esto, fue necesario hacer la diferencia entre plagio y reutilización de texto, encontrando textos completos idénticos al texto original o únicamente pasajes utilizados sin haber sido citados. Conociendo esto, decidimos que el software desarrollado fuera capaz tanto de reconocer plagio como reutilización de texto.
Para esto, realizamos el preprocesamiento de los textos, incluyendo la separación y limpieza mediante lematización y la eliminación de signos de puntuación de estos, para posteriormente realizar una comparación del texto sospechoso y el texto original con ayuda de n-gramas. Una vez realizada dicha comparación, obtendremos como resultado una lista con el porcentaje de similitud y la ocurrencia entre ambos textos, con lo que podremos determinar si el texto sospechoso se trata o no de reutilización de texto.

2. Abstract

The detection of plagiarism and reuse of text is of paramount importance today, as falling into this is committing a crime. This is why various techniques have been developed to detect it. In this document, we will develop and describe the implementation of the software developed to achieve the detection of both plagiarism and reuse of text. To achieve this, it was necessary to make the difference between plagiarism and reuse of text, finding complete texts identical to the original text or only passages used without having been cited. Knowing this, we decided that the software developed was capable of both recognizing plagiarism and reusing text.

For this, we perform the preprocessing of the texts, including the separation and cleaning by lemmatization and the removal of punctuation marks from these, to later make a comparison of the suspicious text and the original text with the help of n-grams. Once such a comparison is made, we will obtain as a result a list with the percentage of similarity and the occurrence between both texts, with which we will be able to determine whether the suspicious text is text reuse or not.

3. Introducción

3.1 Descripción de la problemática

En la actualidad, las leyes de derechos de autor protegen muchos tipos de diferentes obras como son pinturas, fotografías, ilustraciones, composiciones musicales, grabación de sonido, programas de computadora, libros, poemas, publicaciones de blogs, películas, obras arquitectónicas, obras de teatro y mucho más. Es por esto que la protección de los derechos de autor es de suma importancia.

La detección de plagio es el proceso de localizar instancias de plagio o infracción de derechos de autor dentro de un trabajo o documento. Sin embargo, cuando se da crédito al trabajo o se emplea un entrecomillado, se puede convertir en una referencia. Es por ello que en también se conoce como detección de similitud de contenido o reutilización de texto. Esta situación escala a diversos ámbitos como el desarrollo de software. Existen herramientas de detección de similitud de código permiten determinar qué tan similar es el código fuente de diferentes desarrolladores. Desarrollar herramientas computacionales que abarquen todos los tipos es muy complejo y demandante en tiempo. Por eso, nos enfocaremos en el reto al desarrollo de herramientas para la detección de infracciones en texto.

En el desarrollo de nuestra solución, implementaremos una herramienta para poder identificar la reutilización de texto, que se basará en encontrar si el texto analizado tiene un porcentaje alto de similitud con el texto original, identificando así si el texto sospechoso incurre en la reutilización de texto.

Con este desarrollo, generamos la siguiente hipótesis:

El software desarrollado podrá detectar 90% del texto reutilizado de una fuente original en otro texto plano generado con ayuda de diferentes técnicas como lematización, n-gramas, similitud de coseno y distancia entre párrafos, que en un inicio será probado con textos extraídos de los recursos de la biblioteca digital del Tecnológico de Monterrey.

3.2 Enfoque de la solución

Tomando en cuenta lo anterior, nuestro reto y actividades diseñamos y desarrollamos, una herramienta de detección de reutilización de texto, que utiliza diversas técnicas de Aprendizaje Automático, Métodos Cuantitativos y Compiladores, para producir una solución efectiva y eficiente que pueda proyectarse como un servicio en línea. Para determinar lo anterior, compararemos la solución propuesta con una lectura de textos y documentos dentro de la base de datos realizada por activadas anteriores.

En la solución del problema, se deberá aplicar técnicas vistas en clase, u otras que sean apropiadas, tales como la subsecuencia común más larga, entre otras. Inicialmente, utilizaremos la lematización, la generación de n-gramas y la similitud de coseno para poder realizar las comparaciones entre los textos genuinos y sospechosos.

En la revisión de literatura relacionada, se deberá demostrar competencia en el uso de bases de datos bibliográficos , disponibles en la biblioteca para identificar los métodos más destacados en relación con el problema.

4. Trabajo relacionado

4.1 Funciones principales

4.1.1 Resumen de la Arquitectura

La solución se divide en tres etapas principales:

- Etapa de preprocesamiento

  En esta etapa, nos encargamos de la lectura, la limpieza y separación de los textos para que estén listos para ser analizados.

- Etapa de preparación

  En esta etapa, nos encargamos de la creación de n-gramas de los textos sospechosos y genuinos, para poderlos comparar posteriormente.

- Etapa de toma de decisiones

  En esta etapa nos encargamos de comparar los resultados de las tapas anteriores para comparar la similitud de ambos textos y, con base en las métricas establecidas con la similitud de coseno y distancia entre párrafos, definiremos si el texto sospechoso incurre en reutilización de texto.

4.1.2 Especificación de la etapa de preprocesamiento

4.1.2.1 Lectura de textos

En la etapa de preprocesamiento es donde nos encargamos de la lectura de los textos sospechosos y genuinos incluidos en el dataset desde una carpeta local.

4.1.2.2 Stemming

Además de abrir los archivos, convertimos todos los textos a letras minúsculas y realizamos una tokenización de las palabras para separarlas por medio de los espacios en blanco. Llevamos a cabo el stemming del texto, con ayuda del LancasterStemmer siendo el más agresivo para encontrar las palabras raíces y realizando de manera automática la limpieza de los textos con funciones anteriores que apoyan a obtener dichas palabras en gracias a este método.

4.1.2.3 Lematización

Por otro lado, además de realizar stemming, también en esta solución, generamos la tokenización de las palabras por medio de los espacios y llevamos la búsqueda de palabras raíces pero a diferencia del anterior método, la lematización la realizamos por medio de reglas, las cuales la llevamos a cabo para poder encontrar la raíz únicamente de las palabras que son verbos, sustantivos y adjetivos que fueron

previamente identificados. Con la lematización, se eliminan los afijos morfológicos, identificando así las raíces de cada una de las palabras. Una vez los textos lematizados, guardamos en una lista las palabras raíces por oración y las palabras raíces por párrafo. Además, generamos una lista que contiene una única raíz, sin repeticiones.

4.1.2.4 Limpieza de textos

Posteriormente, realizamos la limpieza de los textos, con el fin de no aumentar el porcentaje de similitud entre dos textos por elementos que se encuentran de manera recurrente en todos o la mayoría de los textos. En nuestro caso, eliminamos los signos de puntuación e ignoramos tanto las expresiones unidas a un símbolo de arroba como enlaces que comienzan por http o https.

4.1.2.5 Resultado

Después de este procesamiento, obtenemos como resultado el texto en minúsculas, listas del texto separado por oraciones, la lista de los textos modificados por Stemming y finalmente una lista de los textos procesados con lematización y limpios de acuerdo con las reglas de eliminación de signos de puntuación y enlaces establecidas, además de una lista con las palabras únicas por frase y por párrafo. Este procedimiento lo llevamos a cabo tanto en los textos genuinos como en los textos sospechosos.

4.1.3 Especificación de la etapa de preparación

4.1.3.1 Textos preprocesados

En esta etapa, tanto los textos genuinos como sospechosos deben estar ya abiertos y preprocesados con los pasos realizados en la etapa previa.

4.1.3.2 N-gramas

Se generan n-gramas con el fin de analizar las palabras clave dentro de un contexto, recorriendo las oraciones y contar la frecuencia en la que estas aparecen dentro de un texto. Se generan secuencias lineales de diversos tamaños, que en nuestro caso son unigramas, bigramas y trigramas, para identificar palabras clave dentro del contexto en el que se encuentra. Las oraciones se separan en grupos de palabras, en el caso de los unigramas, los grupos constan de solo una palabra, en los bigramas son grupos de dos palabras y finalmente en el caso de los trigramas son grupos de tres palabras. Generamos los n-gramas tanto de los textos genuinos como de los textos sospechosos.

4.1.3.3 Resultados

En esta sección, obtenemos como resultado los n-gramas de los textos previamente limpios tanto genuinos como sospechosos.

4.1.4 Especificación de la etapa de toma de decisiones

4.1.4.1 Textos preprocesados

Para esta etapa, utilizamos los textos de las etapas precedentes, tanto los textos originales como los textos divididos con n-gramas. En el caso de los textos preprocesados, conservamos únicamente una lista con las palabras, para no trabajar sobre tuplas.

4.1.4.2 Similitud de coseno

Para calcular la similitud entre ambos textos, calculamos la similitud de coseno entre dos vectores pertenecientes a un texto genuino y un texto sospechoso. Utilizamos un texto sospechoso y el dataset de los textos genuinos con el fin de que el texto sospechoso se pueda comparar con cada uno de los textos genuinos pertenecientes al dataset, obteniendo como resultado una matriz con números de 0 a 1, siendo el más alto aquel con el que tiene mayor similitud, el cual nos permite determinar si el texto incurre en reutilización de texto.

4.1.4.3 Resultados

Posteriormente al cálculo de la similitud de coseno, obtenemos por cada texto sospechoso una matriz conteniendo N resultados de similitud de coseno, correspondiente al número N de textos genuinos incluidos en el dataset.

4.2 Dependencias de la solución

Para la implementación de esta solución, utilizamos las siguientes dependencias:

- re: Nos permite realizar operaciones de coincidencia de expresiones regulares.
- os: Nos permite abrir y leer los documentos de texto plano.
- nltk: Esta librería nos permite realizar análisis de texto por medio del procesamiento de lenguaje natural.
    - tokenize -> word_tokenize: Este módulo permite dividir un string en una lista de substrings de acuerdo con los espacios en blanco y los signos de puntuación.
    - stem -> WordNetLemmatizer: Este módulo nos permite eliminar los afijos de las palabras, conservando únicamente la raíz de la palabra.
    - tag -> pos_tag: Este módulo permite identificar y clasificar el tipo de palabra del que se trata (adjetivo, verbo, sustantivo, entre otros).
    - util -> ngrams: Este módulo permite realizar los n-gramas (tanto unigramas, bigramas, trigramas, entre otros).
- sklearn: Nos permite implementar modelos de aprendizaje máquina y modelos estadísticos.
    - CountVectorizer: Este módulo permite convertir un texto a una matriz de tokens.
    - cosine_similarity: Este módulo permite calcular la similitud de coseno entre dos entradas, X y Y.

4.3 Marco experimental

En esta sección describiremos nuestro marco experimental. Para abordar la problemática, decidimos realizar la limpieza de los textos para posteriormente llevar a cabo tanto la lematización como el stemming de los textos, y generación de n-gramas. La limpieza de los textos la realizamos con ayuda de expresiones regulares y reemplazamiento de caracteres. Además, llevamos a cabo la lematización y el Stemming con ayuda del Lancaster Stemmer, con el fin de obtener las palabras raíces de cada texto (aste proceso está disponible en

https://github.com/Lalcosta/PlagiarismDetectorTeam4.git , accesado el 28 de mayo de 2023).

4.3.1 Análisis de datasets

Para la elaboración de nuestra solución, utilizamos un dataset conteniendo 120 documentos de texto plano genuinos, que fueron comparados con 33 documentos de texto plano conteniendo texto de otros archivos y 15 documentos de texto plano sospechosos. Todos los textos genuinos fueron extraídos de textos relacionados a la detección de plagio, encontrados en los recursos digitales de la Biblioteca del Tecnológico de Monterrey. Los textos conteniendo otros archivos contienen textos de otras fuentes dentro de los textos genuinos. Finalmente, los textos sospechosos son los textos genuinos con distintas palabras, tiempos verbales y estructuras gramaticales modificadas.

4.3.2 Protocolo de evaluación

Nuestro protocolo de evaluación fue de la siguiente manera. Para iniciar, realizamos la limpieza de los textos. Posteriormente, utilizamos Lancaster Stemmer para encontrar las palabras raíz del texto. A la par, realizamos la lematización de los textos limpios para, de igual manera, encontrar la raíz de las palabras clave del texto.
Posteriormente, utilizamos la técnica de los n-gramas que nos ayudó a encontrar las palabras clave dentro de un contexto.
Después de este proceso, calculamos la similitud de coseno, que mide la similitud entre dos vectores distintos. Esto nos dará un valor entre 0 y 1. Dentro de este rango, definimos que si los resultados eran mayores a 0.9, el texto incurre en reutilización de texto. Un valor menor a este rango, es un texto genuino.

4.4 Análisis de Resultados

El software desarrollado utiliza diferentes técnicas de limpieza, separación y análisis de similitud de los textos.
En nuestro caso, decidimos utilizar la lematización con limpieza de texto y el stemming con el fin de comparar cuál de los dos métodos realiza el preprocesamiento de textos de manera más eficiente y correcta, como también su dicha comparación de similitudes con los n-gramas generados durante la limpieza de los textos.

Ahora bien, tras ya tener los documentos limpios y generados, estos, en formato de listas, fueron vectorizados y comparados por el método de similitud de cosenos, la cual una vez generado el proceso, se analizó y se juzgó un documento sospechoso con cada uno de los documentos genuinos y por cada método, dando el Top 3 de similitudes por cada documento sospechoso santos lo siguientes resultados:

Bigramas

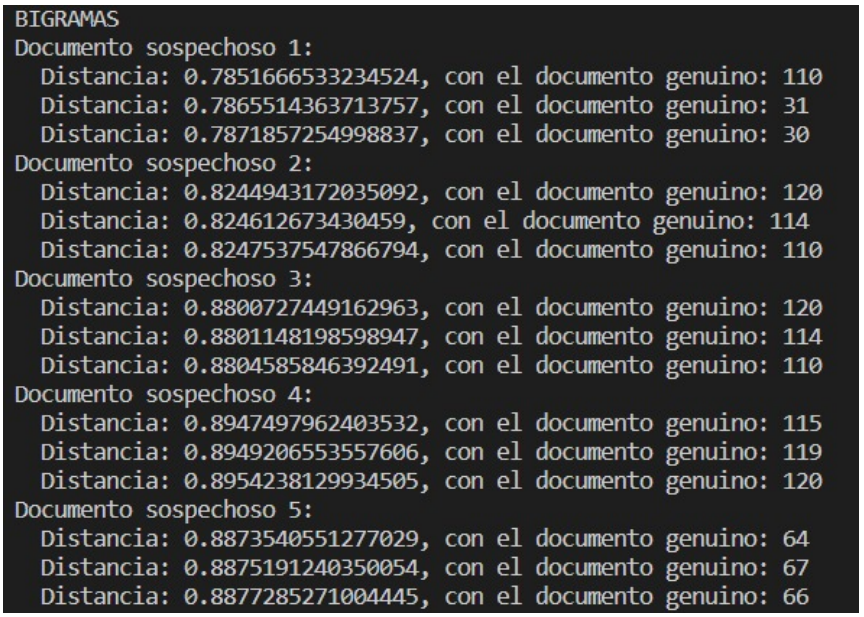

Imagen 1: Resultado de Bigramas

Trigramas

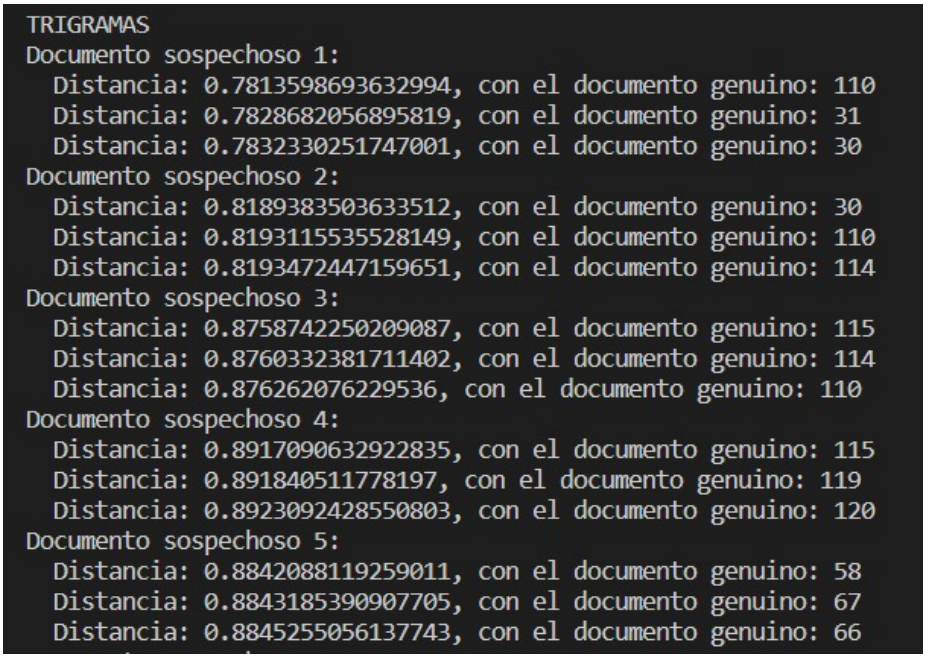

Imagen 2: Resultado de Trigramas

Lematización

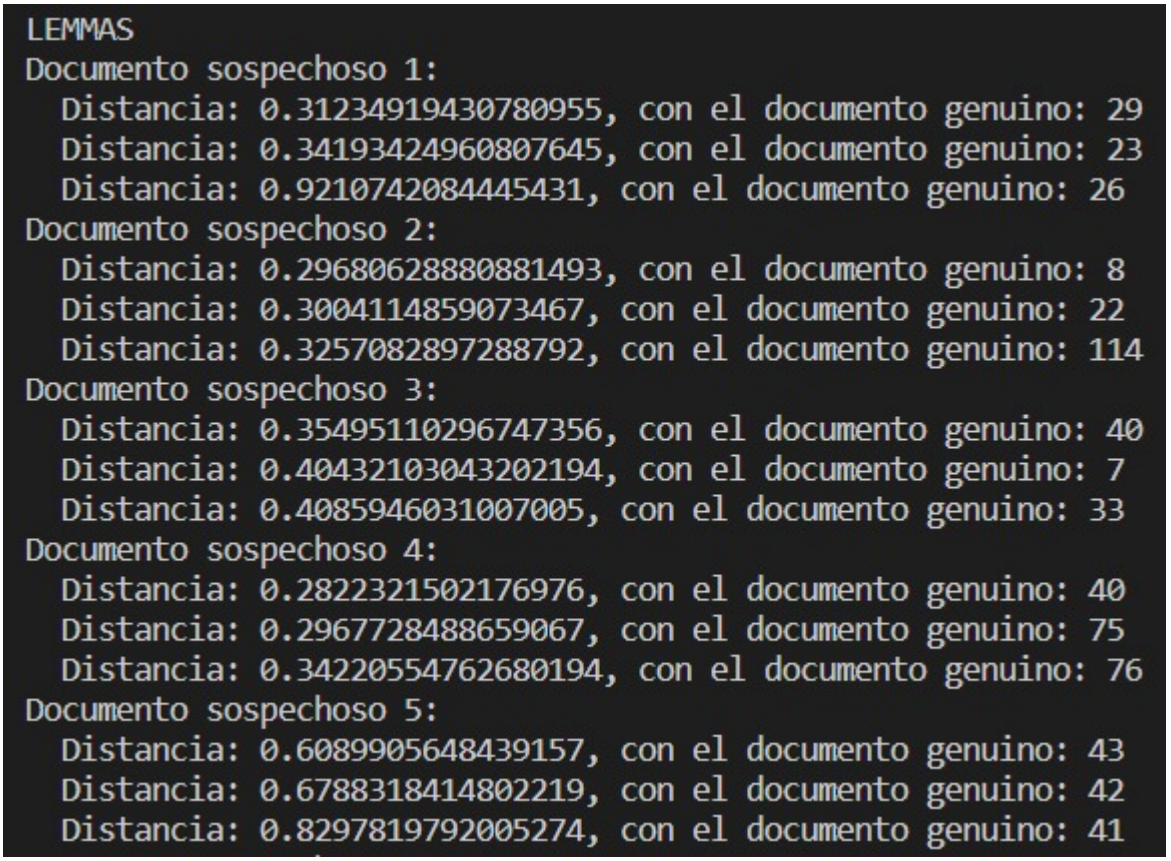

Imagen 3: Resultado de Lematización

Stemming

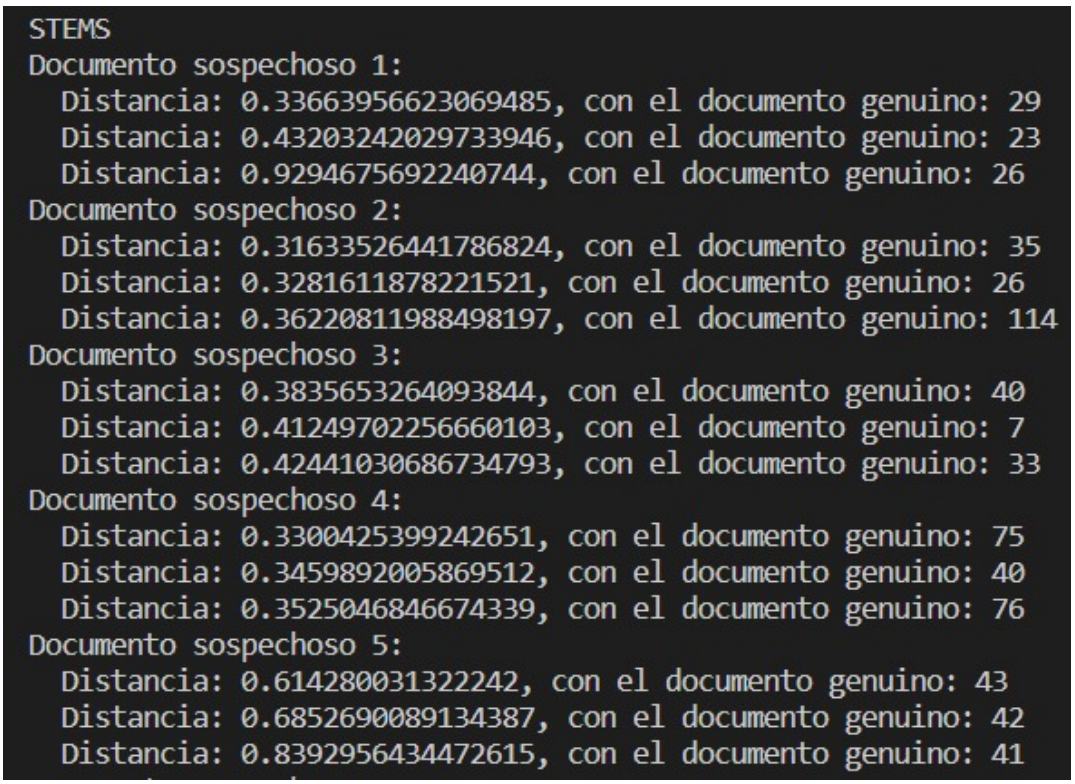

Imagen 4: Resultado de Stemming

Ahora bien, tras ver los resultados generados por la pruebas creadas, podemos destacar que, tanto el método de Stemming y Lematización son más precisos y eficientes al momento de analizar un documento sospechoso contra los documento genuinos, ya que, tras juzgar un documento sospechoso con un documento genuino y por las palabras raíces generadas por ambos métodos, nos da un acercamiento más realista y precisa al momento de juzgarlo, dando los resultados esperados al momento de checar la similitud y el uso de texto utilizado en el mismo. Lo anterior se puede observar en las imágenes 3 y 4 donde si hay un documento con una distancia de coseno predominante.

Por otro lado, al ver la imagen 1 y 2, al usar trigramas como bigramas, estos dan a conocer precisiones y similitudes entre documento muy parecidas, lo cual nos da a conocer que al usar estos métodos de limpieza y detección puede arrojar falsos positivos debido a que las repeticiones constantes de palabras provoca que los textos de parezcan más

5. Conclusiones

En el documento presentado realizamos una experimentación acerca de la detección de reutilización de texto utilizando diferentes herramientas computacionales. Para lograr la detección de reutilización de texto, realizamos el proceso en tres etapas, empezando por la limpieza de los textos que nos permitió eliminar los símbolos y expresiones existentes en en todos o la mayoría de los archivos. También llevamos a cabo con este proceso la lematización de los archivos, obteniendo las palabras raíces. Por otro lado, sobre los textos originales, llevamos a cabo el stemming, que realiza tanto la limpieza de los textos como la obtención de palabras raíces de manera más profunda. En nuestra segunda etapa, llevamos a cabo la generación de n-gramas. útiles para encontrar las palabras clave dentro de un contexto de n-palabras, recorriendo toda la lista de palabras. Finalmente, con los resultados generados en las etapas anteriores, realizamos la comparación de similitud por medio de la similitud de coseno de un texto sospechoso con todos los textos genuinos contenidos en el dataset, para poder obtener el texto con el que tiene mayor similitud y cae en la reutilización de texto. Con este proceso, logramos detectar de manera exitosa y efectivamente la reutilización de texto en un porcentaje alto al utilizar el método de Stemming y Lematización gracias a su acercamientos realistas y específicos de cada documento comparado con los datos genuinos presentes en el proyecto a comparación con los bigramas y trigramas , ya que, al tener una alta proporción de repetición dentro de la lista creada para cada documento, esta nos daba falso positivos en la mayoría de los documentos analizados con los documentos genuinos, lo

cual nos lleva a destacar el acercamiento y precision del metodo Stemming y Lematización, gracias a la reglas que se realizaron durante la limpieza de textos de cada uno de los documento su análisis por similitud de cosenos una vez ya juzgado y analizado un documento sospechosos contra los documento genuinos.

6. Referencias

Joshi, S., & Banerjee, I. (2023). *Ngram-LSTM Open Rate Prediction Model (NLORP) and Error_accuracy@C metric: Simple effective, and easy to implement approach to predict open rates for marketing email.*

Toporkov, O., & Agerri, R. (2023). *On the Role of Morphological Information for Contextual Lemmatization.*

Srivastava, R. P. (2023). A New Measure of Similarity in Textual Analysis: Vector Similarity Metric versus Cosine Similarity Metric. *Journal of Emerging Technologies in Accounting, 20(1), 77–90.* *https://0-doi-org.biblioteca-ils.tec.mx/10.2308/JETA-2021-043*

Shah, J. N., Shah, J., Baral, G., Baral, R., & Shah, J. (2021). Types of plagiarism and how to avoid misconduct: Pros and cons of plagiarism detection tools in research writing and publication. *Nepal Journal of Obstetrics & Gynaecology, 16(2), 3–18.* *https://0-doi-org.biblioteca-ils.tec.mx/10.3126/njog.v16i2.42085*

Tipos-de-Plagio. (s. f.). https://diegoliveros.github.io/PactoHonor/tipos-de-plagio.html

Turkel, W. J. (2012, 17 julio). *Palabras clave en contexto (usando n-grams) con Python*. Programming Historian. https://programminghistorian.org/es/lecciones/palabras-clave-en-contexto-n-grams

Stemming and lemmatization. (s. f.). https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html