Instituto Tecnológico y de Estudios Superiores de Monterrey
Campus Monterrey

Desarrollo de aplicaciones avanzadas de ciencias computacionales
TC3002B, Grupo 301

Raúl Monroy
Jorge Adolfo Ramírez Uresti
Ariel Ortiz Ramírez
Miguel González Mendoza

Evidencia 2 - Fase 3 - Implementación de IA

Equipo #2 | Integrantes:

Germán Guzmán López	A01752165
Isabel Vieyra Enríquez	A01745860
Marco Barbosa Maruri	A01746163

Mayo 2024

Índice

Demostración del Mérito Relativo de las Herramientas Desarrolladas

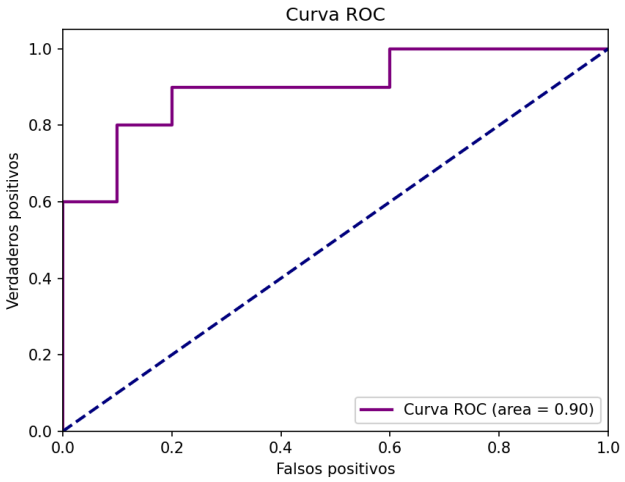
En este apartado, se compara el desempeño de dos versiones de una herramienta de detección de plagio desarrolladas durante el curso. La primera versión de la herramienta utiliza lematización, embeddings y la distancia del coseno para determinar si existe un posible plagio a partir de cierto umbral.

Resultados de la Primera Versión

Para evaluar el rendimiento de esta versión, se realizó una serie de pruebas que arrojaron un accuracy del 81%. Además, se construyó una matriz de confusión binaria cuyos resultados se muestran a continuación:

Falsos Positivos (FPR)	Verdaderos Positivos (TPR)
0.0	0.0
0.0	0.1
0.0	0.6
0.1	0.8
0.2	0.8
0.2	0.9
0.6	0.9
0.6	1.0
1.0	1.0

Utilizando estos datos, se calculó el área bajo la curva (AUC) obteniendo un valor de 0.9.



Este resultado indica que el modelo tiene un desempeño muy bueno, evidenciando una alta capacidad para separar documentos plagiados de los originales. Cabe destacar que un AUC de 1.0 representa un modelo perfecto, mientras que un AUC de 0.5 indica un desempeño equivalente a realizar una selección al azar.

Además, se determinó que el umbral óptimo basado en la métrica F1 es 0.19, con una precisión de 0.81, un recall de 0.9 y un F1 score de 0.85. Estos valores reflejan un excelente equilibrio entre la precisión y la capacidad de detección del modelo.

Para demostrar el mérito relativo de esta herramienta frente a una segunda versión, se comparan diversos aspectos, tales como la precisión (accuracy), la capacidad de discriminación (AUC), el valor de la metrica F1 y el recall, y la robustez frente a diferentes tipos de datos. La segunda versión, cuya implementación y resultados específicos se describirán más adelante, se evalúa bajo los mismos criterios y pruebas.

La primera versión, con la combinación de lematización, embeddings y distancia del coseno, demuestra un excelente equilibrio entre precisión y capacidad de discriminación, justificando su implementación inicial y proporcionando una base sólida para posteriores mejoras que se tienen previstas y comparaciones.

Resultados de la Segunda Versión

La segunda versión de la herramienta utiliza técnicas avanzadas de IA para mejorar la detección de plagio. Los resultados obtenidos para esta versión se desglosan en dos medidas principales: detección de plagio (sí o no) y clasificación del tipo de plagio.

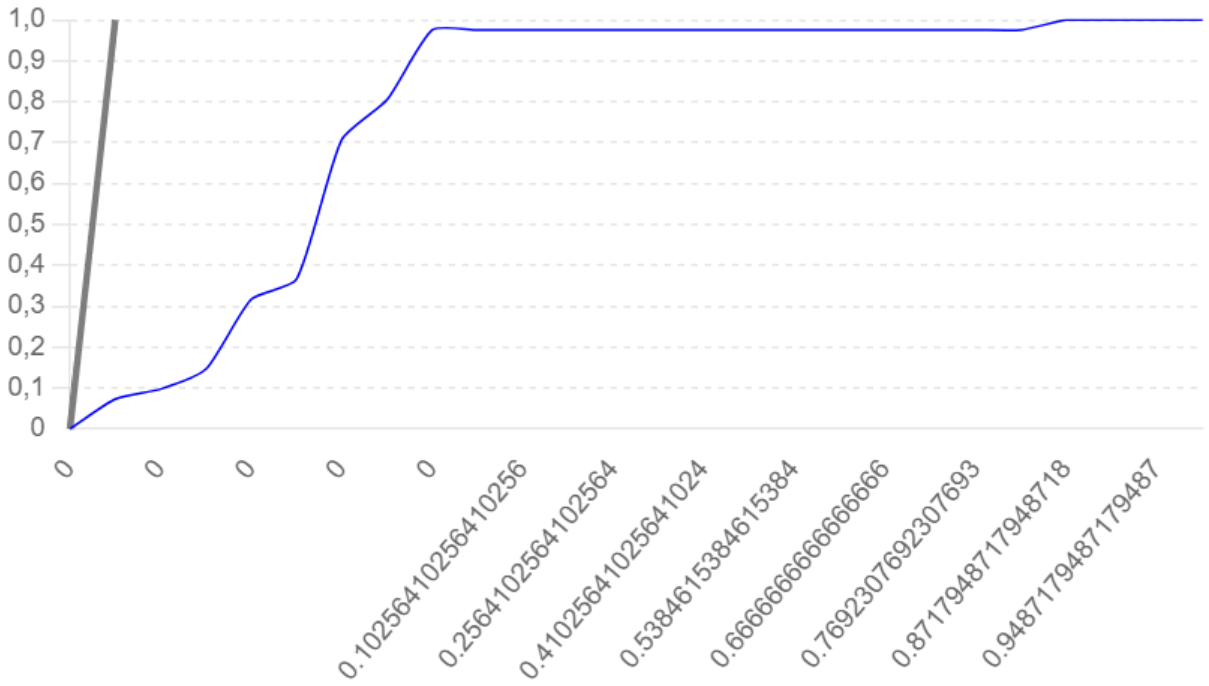
Plagio/No Plagio:

F1 Score Global: 1.00
Precisión Global: 1.00
Recall Global: 1.00
Accuracy Global: 1.00

Tipo de Plagio:

F1 Score Global: 0.43
Precisión Global: 0.45
Recall Global: 0.44
Accuracy Global: 0.43

Estos resultados reflejan un desempeño excepcional en la detección de si un documento es plagiado o no, con una precisión y recall perfectos (1.00), lo que indica que la herramienta no comete errores en esta clasificación. El umbral óptimo para la detección de plagio es 0.43, y el área bajo la curva (AUC) es 0.97, lo que subraya una excelente capacidad de discriminación.



Sin embargo, la clasificación del tipo específico de plagio muestra un desempeño moderado, con un F1 Score, precisión y recall alrededor de 0.43-0.45, lo cual sugiere áreas para futuras mejoras.
Más adelante se mencionan con detalle los resultados obtenidos con esta segunda versión.

Comparación

Comparando ambas versiones, la segunda versión muestra una capacidad superior para detectar plagio en general, alcanzando una precisión y recall perfectos. Esto representa una mejora significativa en comparación con la primera versión, que aunque robusta y efectiva, no alcanzaba la perfección en su desempeño.

No obstante, cuando se trata de identificar el tipo específico de plagio, la segunda versión aún tiene margen para mejoras, con métricas que indican una clasificación moderada. Esta comparación destaca la importancia de utilizar técnicas avanzadas de IA para mejorar la precisión general, al mismo tiempo que subraya la necesidad de seguir refinando los modelos para tareas más detalladas.

Entonces, la segunda versión demuestra un mérito relativo considerable, especialmente en la detección binaria de plagio, estableciendo un nuevo estándar de precisión y recall en nuestra herramienta de detección de plagio.

Nueva Relación de Similitud Propuesta Basada en IA

Para este nuevo modelo, la relación de similitud está basada en la capacidad de BERT para analizar el contenido semántico de los textos. BERT utiliza una arquitectura de transformers, que permite capturar relaciones contextuales complejas en el texto asignando pesos a las palabras más relevantes de una oración en función de la importancia de cada palabra mediante mecanismos de atención conocidos como 'cabezas'. Estas cabezas se basan en vectores de 'query 'y 'key', que calculan la compatibilidad entre pares de palabras y generan puntuaciones de atención. Estas puntuaciones se normalizan mediante la función softmax para obtener pesos, los cuales indican la relevancia de cada palabra en relación con las demás. Este enfoque proporciona una comprensión profunda del contexto y las relaciones semánticas en el texto para identificar patrones significativos y sutiles de similitud, mejorando la precisión de la relación de similitud propuesta anteriormente, como la similitud coseno.

Justificación de la Elección de BERT

BERT fue elegido debido a su capacidad para entender y procesar el contexto de las palabras en una oración, lo que es crucial para la detección precisa de plagio. A diferencia de técnicas más simples como la lematización y los embeddings estáticos, BERT puede capturar la variabilidad semántica y contextual de las palabras, lo cual es esencial para identificar similitudes textuales que no son evidentes a simple vista.

Aplicación de la Relación de Similitud

La relación de similitud se aplica sobre los vectores generados por BERT para los textos en cuestión. BERT transforma cada palabra del texto en un vector de alta dimensión que representa su contexto semántico. Luego, se compara la similitud entre estos vectores utilizando técnicas como la distancia euclidiana o el coseno de los ángulos entre los vectores, aunque el enfoque más común es usar la salida del token [CLS] de BERT para representar el texto completo.

1. Vectores de Contenido: Los vectores utilizados para la comparación son los obtenidos del token [CLS] de BERT, que captura el significado contextual del texto completo.
2. Comparación de Vectores: La similitud entre dos textos se determina comparando los vectores [CLS] correspondientes. Para textos A y B, la similitud se calcula utilizando la fórmula de similitud del coseno:

$$Similitud(A, B) = \frac{A \cdot B}{||A||\,||B||}$$

3. Umbral de Decisión: Se fijó un umbral de 0.43 para decidir si un texto es considerado plagio o no. Este umbral se determinó mediante la optimización del F1 Score durante la fase de evaluación, buscando el punto que mejor equilibrara precisión y recall para la tarea de detección de plagio.

Determinación del Umbral

El umbral del 0.43 se obtuvo a partir de un análisis detallado del rendimiento del modelo en un conjunto de validación. Este valor se seleccionó porque maximiza la métrica F1, que es el balance entre precisión y recall.

Selección y Aplicación de Técnicas de IA en la Implementación de la Herramienta

Selección

Para la implementación de la herramienta final de detección de plagio, se evaluaron tres técnicas de inteligencia artificial con el objetivo de optimizar la precisión y efectividad de la detección.

Evaluación de opciones

Inicialmente, se exploraron tres técnicas de procesamiento de lenguaje natural y modelos de aprendizaje automático para la tarea de detección de plagio, incluyendo:

- **BERT**: Un modelo basado en transformers que permite la comprensión bidireccional del contexto, mejorando significativamente la precisión en la detección de plagio al considerar tanto el contexto anterior como posterior de las palabras.
- **DistilBERT**: Una versión más ligera y rápida de BERT que, si bien ofrece tiempos de inferencia más rápidos, no alcanza el mismo nivel de precisión en tareas complejas.
- **Random Forest**: Un método de aprendizaje supervisado basado en conjuntos de árboles de decisión, conocido por su capacidad para manejar grandes conjuntos de datos y resistir el sobreajuste, pero limitado en su capacidad para comprender el contexto semántico profundo de los textos.

Resultados y Comparación

Tras implementar y probar cada técnica, se obtuvieron los siguientes resultados para **BERT**:

	Precisión	Recall	F1-Score	Support
Originales	1.00	0.57	0.73	6
Insertar frases	0.38	0.43	0.40	7
Reemplazar frases	0.14	0.17	0.15	6
Cambio de voz	0.20	0.14	0.17	7
Desordenar frases	0.33	0.29	0.31	7

Parafraseo	0.38	0.50	0.43	6
Cambio de tiempo	0.75	1.00	0.15	6
Accuracy			0.43	46

	Precisión	Recall	F1-Score	Support
Plagio	1.00	1.00	1.00	39
No Plagio	1.00	1.00	1.00	38
Accuracy			1.00	77

DistilBERT:

	Precisión	Recall	F1-Score	Support
Originales	0.56	1.00	0.71	5
Insertar frases	0.25	1.00	0.40	4
Reemplazar frases	0.00	0.00	0.00	3
Cambio de voz	0.00	0.00	0.00	3
Desordenar frases	0.00	0.00	0.00	4
Parafraseo	0.00	0.00	0.00	3
Cambio de tiempo	0.00	0.00	0.00	3
Accuracy			0.36	25

Random Forest:

	Precisión	Recall	F1-Score	Support
Originales	0.82	1.00	0.90	9
Insertar frases	0.00	0.00	0.00	7

Reemplazar frases	0.20	0.14	0.17	7
Cambio de voz	1.00	0.43	0.60	7
Desordenar frases	0.57	0.57	0.57	7
Parafraseo	0.50	0.14	0.22	7
Cambio de tiempo	0.00	0.00	0.00	6
Accuracy			0.36	50

Estos resultados fueron comparados entre sí, donde BERT mostró un desempeño superior, especialmente en la tarea de detección de tipo de plagio. Por lo tanto se tomó la decisión de descartar las implementaciones con DistilBERT y Random Forest.

Aplicación

Para la implementación final de la herramienta se seleccionó BERT como la técnica principal de inteligencia artificial. A continuación, se describe en detalle el proceso de aplicación de BERT en el proyecto:

1. Importación de librerías y definición de funciones
    - Se importaron las librerías necesarias para el procesamiento de datos, el entrenamiento del modelo y la evaluación de resultados. Además, se definieron funciones auxiliares como *read_text_from_file* y *ensure_txt_extension* para manejar los archivos de texto y asegurar las extensiones correctas.
2. Carga y preprocesamiento de datos
    - Los datos se cargaron desde un archivo CSV (Training.csv) y se procesaron para incluir el texto de los documentos sospechosos.
    - Se separa en dos subprocesos:
        - Lectura de Archivos de Texto: Se leyeron los textos de los archivos sospechosos.
        - Mapeo de Etiquetas: Se crearon etiquetas numéricas para los diferentes tipos de plagio utilizando *label_dict*.
3. Asegurar un mínimo de ejemplos por clase
    - Mediante la función *ensure_min_class_size*, se aseguró que cada clase tuviera al menos dos ejemplos, esto para buscar garantizar la validez del entrenamiento.
4. Balanceo de datos

- Se realizó un balanceo de datos mediante el remuestreo (*resample*) para asegurar que todas las clases estuvieran representadas de manera equilibrada en el conjunto de datos.

5. División de datos
   - Los datos se dividieron en conjuntos de entrenamiento, validación y prueba utilizando *train_test_split*, asegurando que cada clase estuviera adecuadamente representada en cada conjunto.
6. Tokenización
   - Se utilizó el tokenizador de BERT (*BertTokenizer*) con el modelo pre entrenado bert-base-uncased para convertir los textos en tokens que el modelo BERT pudiera procesar. Este proceso incluyó:
     - Tokenización: Convertir el texto en tokens.
     - Padding: Asegurar que todos los textos tuvieran la misma longitud.
     - Truncamiento: Cortar los textos que fueran demasiado largos.
7. Definición del modelo
   - Utilizamos dos modelos pre entrenados basados en BERT, primero *bert-base-uncased* de la librería Hugging Face. Este modelo ha sido pre entrenado en un gran corpus de texto en inglés, permitiendo capturar el contexto bidireccional de las palabras, lo cual es crucial para entender la semántica y la sintaxis del lenguaje natural.

   - También utilizamos *BertTokenizer* para tokenizar los textos sospechosos y los textos originales, que luego se concatenan utilizando el token separador especial de BERT.. El modelo *BertForSequenceClassification* se ha ajustado para dos tareas principales: la detección de plagio (con etiquetas binarias) y la clasificación del tipo de plagio (*um_labels=len(tipo_plagio_labels)*).

   - Para el entrenamiento y la evaluación del modelo, hemos dividido nuestro conjunto de datos utilizando la clase `PlagiarismDataset` en conjuntos de entrenamiento, validación y prueba. Los datos incluyen tanto los textos sospechosos como los originales, lo que permite al modelo comparar y detectar similitudes y diferencias. Además, hemos equilibrado las clases para asegurar un rendimiento óptimo del modelo en la detección de plagio.
     Ej. Para detección de plagio

```
train_dataset = PlagiarismDataset(
     texts=train_df['texto_sospechoso'].tolist(),

original_texts=train_df['texto_original'].tolist(),
     labels=train_df['plagio'].tolist(),
     tokenizer=tokenizer
)
```

     Ej. Para detección de tipo de plagio

```
train_dataset_tipo = PlagiarismDataset(
     texts=train_df['texto_sospechoso'].tolist(),
```

```
original_texts=train_df['texto_original'].tolist(),
     labels=train_df['tipo_plagio'].tolist(),
     tokenizer=tokenizer)
```

8. Configuración del entrenamiento
    - Se configuraron los argumentos de entrenamiento utilizando *TrainingArguments*, incluyendo parámetros como:
        - Número de Epochs (*num_train_epochs*)
        - Tamaño del Batch (*per_device_train_batch_size* y *per_device_eval_batch_size*)
        - Tasa de Aprendizaje *(learning_rate)*
        - Estrategia de Evaluación y Guardado (*eval_strategy* y *save_strategy*)
9. Entrenamiento del modelo
    - El modelo fue entrenado utilizando el `Trainer`, lo cual facilita la gestión del ciclo de entrenamiento y evaluación, y permite ajustes del modelo pre entrenado a nuestra tarea específica. Al final del proceso, el modelo entrenado es capaz de predecir tanto la probabilidad de plagio como el tipo de plagio, proporcionando una herramienta robusta para la detección y análisis de plagio en textos.

```
model_tipo =
BertForSequenceClassification.from_pretrained('bert-base
-uncased', num_labels=len(tipo_plagio_labels))

trainer_tipo = Trainer(
     model=model_tipo,
     args=training_args,
     train_dataset=train_dataset_tipo,
     eval_dataset=val_dataset_tipo,
     compute_metrics=lambda p:
classification_report(p.label_ids,
p.predictions.argmax(-1), output_dict=True))
```

    - Donde:
        - ***model***: el modelo multiclase *BertForSequenceClassification*.
        - **`args`**: los argumentos de entrenamiento definidos en *training_args*, los cuales incluyen configuraciones como el número de epochs = 5, tamaño del lote por dispositivo = 4 y estrategia de evaluación `eval_strategy="epoch", metric_for_best_model="eval_loss”`.
        - ***train_dataset***: el conjunto de datos de entrenamiento *train_dataset_tipo*, que contiene textos sospechosos y originales tokenizados y etiquetados con el tipo de plagio correspondiente.

- ***eval_dataset***: el conjunto de datos de validación *val_dataset_tipo,* que se utiliza para evaluar el rendimiento del modelo durante el entrenamiento.
- ***compute_metrics***: una función lambda que utiliza *classification_report* para calcular y devolver métricas de evaluación, como precisión, recall y F1-score, a partir de las predicciones del modelo y las etiquetas reales.

10. Evaluación del modelo

- Aquí, el modelo realiza predicciones sobre el conjunto de datos de prueba y se recopilan las etiquetas reales y las predicciones generadas para su posterior evaluación.

```
predictions_tipo, labels_tipo, _ =
trainer_tipo.predict(test_dataset_tipo) preds_tipo =
predictions_tipo.argmax(-1)
```

- Una vez obtenidas las predicciones, se calculó la probabilidad de pertenencia a la clase positiva (plagio) utilizando la función de activación softmax.

```
probs =
torch.nn.functional.softmax(torch.tensor(predictions),
dim=-1).numpy()
```

- Esta línea de código utiliza la función softmax para calcular las probabilidades de las predicciones generadas por el modelo.
- Finalmente, se utilizó un umbral para convertir las probabilidades en predicciones binarias de plagio o no plagio. Esto se hizo con la siguiente línea de código:

```
best_preds = (plagio_probs >= threshold).astype(int)
```

- Aquí, se establece un umbral para decidir si una instancia se clasifica como plagio o no plagio en función de la probabilidad generada por el modelo
- Tras generar predicciones con el modelo entrenado en el conjunto de datos de prueba, se procedió a evaluar su rendimiento. Utilizando métricas de evaluación estándar, como precisión, recall y F1-score, se analizó la capacidad del modelo para clasificar correctamente los textos como casos de plagio o no plagio. Estas métricas fueron calculadas utilizando la función *`classification_report`* de la biblioteca *`sklearn.metrics`*, que compara las etiquetas reales con las predicciones del modelo. Además, se ajustó un umbral para la probabilidad de plagio, y se generaron predicciones binarias basadas en este umbral. Esto permitió determinar cómo el modelo se desempeñó en la detección de casos de plagio, proporcionando una comprensión clara de su eficacia y precisión en esta tarea crucial de la detección de plagio.

11. Predicción y generación de resultados
    - Se realizaron predicciones sobre un conjunto de datos sospechosos, generando una tabla de resultados que incluye:
        - Archivo: Nombre del archivo.
        - Plagio (si / no): Indicación de si el texto es considerado plagio o no.
        - Tipo de plagio: El tipo específico de plagio.
        - Fuente: La fuente del plagio si está disponible.

Este proceso asegura que el modelo BERT esté adecuadamente preparado y evaluado para la tarea específica de detección y clasificación de plagio

Resultados de la Ejecución del Protocolo de Evaluación

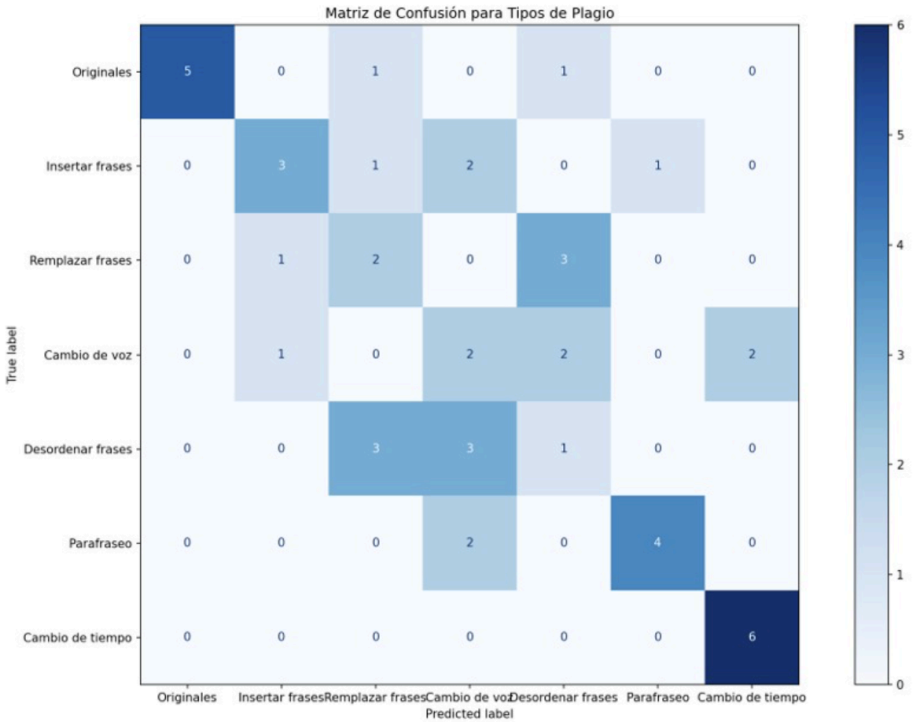
Los resultados obtenidos de ejecutar el protocolo de evaluación con la herramienta de detección de plagio son los siguientes:

Documento sospechoso	Copia	Documento Plagiado	% plagio	Tipo de plagio
FID-001.txt	No	org-017.txt	16.71	Parafraseo
FID-002.txt	No	org-020.txt	24.72	Cambio de voz
FID-002.txt	No	org-020.txt	24.72	Desconocido
FID-003.txt	No	org-060.txt	9.26	Desconocido
FID-003.txt	No	org-060.txt	9.26	Desconocido
FID-005.txt	Sí	org-023.txt	81.16	Originales
FID-007.txt	No	org-002.txt	29.73	Insertar frases
FID-007.txt	No	org-002.txt	29.73	Reemplazar frases
FID-007.txt	No	org-002.txt	29.73	Desconocido
FID-008.txt	No	org-060.txt	17.39	Desconocido
FID-014.txt	Sí	org-019.txt	78.34	Cambio de tiempo
FID-018.txt	Sí	org-057.txt	94.93	Desconocido
FID-028.txt	Sí	org-048.txt	88.61	Parafraseo
FID-046.txt	Sí	org-035.txt	89.66	Desconocido
FID-061.txt	Sí	org-020.txt	64.2	Originales
FID-067.txt	Sí	org-038.txt	70.53	Desconocido
FID-075.txt	Sí	org-098.txt	95.01	Insertar frases
FID-082.txt	Sí	org-047.txt	82.87	Parafraseo
FID-091.txt	Sí	org-078.txt	100	Originales
FID-106.txt	Sí	org-010.txt	46.86	Insertar frases
FID-112.txt	No	org-063.txt	17.74	Desordenar frases
FID-113.txt	No	org-002.txt	24.53	Desconocido

FID-118.txt	No	org-035.txt	27.83	Cambio de voz
FID-123.txt	No	org-060.txt	16.58	Desconocido
FID-125.txt	Sí	org-044.txt	98.2	Cambio de voz
FID-126.txt	No	org-090.txt	12.1	Cambio de tiempo
FID-127.txt	No	org-056.txt	27.01	Cambio de tiempo
FID-127.txt	No	org-056.txt	27.01	Reemplazar frases
FID-127.txt	No	org-056.txt	27.01	Desconocido
FID-128.txt	No	org-056.txt	20.49	Parafraseo
FID-129.txt	No	org-002.txt	19.62	Desordenar frases
FID-129.txt	No	org-002.txt	19.62	Insertar frases
FID-129.txt	No	org-002.txt	19.62	Desconocido
FID-131.txt	No	org-062.txt	13.7	Cambio de tiempo
FID-131.txt	No	org-062.txt	13.7	Desconocido
FID-135.txt	Sí	org-006.txt	87.36	Desconocido
FID-141.txt	Sí	org-067.txt	88.02	Desconocido
FID-145.txt	Sí	org-040.txt	96.12	Cambio de voz
FID-154.txt	Sí	org-003.txt	94.74	Desordenar frases
FID-160.txt	Sí	org-057.txt	96.23	Desordenar frases
FID-165.txt	Sí	org-045.txt	100	Reemplazar frases
FID-170.txt	Sí	org-008.txt	89.33	Insertar frases
FID-171.txt	Sí	org-055.txt	100	Reemplazar frases
FID-178.txt	Sí	org-044.txt	95.01	Desconocido
FID-184.txt	Sí	org-021.txt	87.47	Parafraseo
FID-189.txt	Sí	org-075.txt	82.75	Insertar frases
FID-193.txt	Sí	org-026.txt	74.54	Desconocido
FID-209.txt	Sí	org-017.txt	14.95	Desconocido
FID-210.txt	Sí	org-020.txt	63.95	Desconocido
FID-214.txt	Sí	org-046.txt	88.19	Reemplazar frases
FID-219.txt	Sí	org-078.txt	92.66	Cambio de voz
FID-222.txt	Sí	org-095.txt	96.38	Desordenar frases
FID-233.txt	Sí	org-100.txt	95.14	Desconocido
FID-243.txt	Sí	org-011.txt	87.81	Cambio de tiempo
FID-245.txt	Sí	org-097.txt	97.51	Parafraseo
FID-247.txt	Sí	org-074.txt	97.86	Cambio de tiempo
FID-248.txt	Sí	org-037.txt	95.42	Parafraseo
FID-254.txt	Sí	org-071.txt	93.77	Desconocido
FID-257.txt	Sí	org-087.txt	98.54	Desconocido
FID-274.txt	Sí	org-100.txt	67.8	Insertar frases
FID-275.txt	Sí	org-021.txt	77.42	Insertar frases
FID-282.txt	No	org-084.txt	30.2	Cambio de tiempo
FID-282.txt	No	org-084.txt	30.2	Desconocido
FID-285.txt	No	org-002.txt	23.63	Reemplazar frases

FID-285.txt	No	org-002.txt	23.63	Desconocido
FID-287.txt	No	org-048.txt	31.2	Reemplazar frases
FID-288.txt	No	org-048.txt	26.13	Originales
FID-290.txt	No	org-002.txt	23.52	Parafraseo
FID-290.txt	No	org-002.txt	23.52	Desconocido
FID-290.txt	No	org-002.txt	23.52	Desconocido
FID-291.txt	No	org-005.txt	26.64	Desordenar frases
FID-292.txt	No	org-082.txt	43.02	Desconocido
FID-294.txt	No	org-040.txt	25.34	Cambio de tiempo
FID-294.txt	No	org-040.txt	25.34	Desconocido
FID-295.txt	No	org-048.txt	28.07	Desconocido
FID-298.txt	No	org-020.txt	28.38	Desconocido
FID-299.txt	No	org-072.txt	25.87	Desconocido

Matriz de confusión por tipos de plagio



Liga del repositorio de GitHub y modelos entrenados

Repositorio: https://github.com/GermanGuzmanLopez/HerramientaDeteccionPlagio.git

Modelos entrenados:
https://drive.google.com/file/d/1S_QL237AdBZuiHDRer7GUIarRcJtaF22/view?usp=sharing

Fuentes

- Unmasking BERT: The Key to Transformer Model Performance. (2023, August 18). neptune.ai. Retrieved June 5, 2024, from https://neptune.ai/blog/unmasking-bert-transformer-model-performance
- Devlin, J., & Chang, M. (2018, November 2). Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing. Google Research. Retrieved June 6, 2024, from https://research.google/blog/open-sourcing-bert-state-of-the-art-pre-training-for-natural-language-processing/
- Introduction to DistilBERT in Student Model. (2022, November 11). Analytics Vidhya. Retrieved June 6, 2024, from https://www.analyticsvidhya.com/blog/2022/11/introduction-to-distilbert-in-student-model/
- What Is Random Forest? (n.d.). IBM. Retrieved June 5, 2024, from https://www.ibm.com/topics/random-forest