Tecnicatura Universitaria en Inteligencia Artificial

Trabajo Práctico Parte 2

Asignatura: Procesamiento del Lenguaje Natural

Profesores: Juan Pablo Manson, Alan Geary, Constantino Ferrucci y Dolores Sollberger

Alumno: Bravi Eugenio

Fecha de entrega: 21/05/2025

Indice

Resumen

A lo largo de los diferentes ejercicios realizados, se han obtenido las siguientes conclusiones principales:

- **Búsqueda Semántica (Ejercicio 2):** Al comparar diferentes métricas para encontrar fragmentos de texto semánticamente similares a una consulta (utilizando representaciones vectoriales densas), la **similitud coseno** demostró ser la más efectiva para capturar la relevancia del significado, superando a las distancias euclidiana y Manhattan en este contexto.
- **Búsqueda Basada en Sustantivos (Ejercicio 3):** Para encontrar documentos relevantes basándose en los sustantivos que comparten con una consulta, la combinación de **TF-IDF sobre los sustantivos lemmatizados y la similitud coseno** ofreció resultados más robustos y significativos que la similitud de Jaccard, especialmente cuando las consultas eran cortas.
- **Detección de Idioma (Ejercicio 4):** Se confirmó la viabilidad de detectar automáticamente el idioma principal de los documentos. Esta capacidad es fundamental para aplicar herramientas de procesamiento de lenguaje natural específicas de cada idioma en etapas posteriores, como la selección del modelo correcto para análisis morfosintáctico o de entidades.
- **Análisis de Sentimientos y Búsqueda Combinada (Ejercicio 5):** Los modelos pre-entrenados para análisis de sentimientos pueden clasificar eficazmente la polaridad de las reseñas (positiva, neutra, negativa). Al combinar esta clasificación con la búsqueda de similitud semántica, es posible encontrar reseñas que no solo traten temas similares a una consulta, sino que también compartan la misma carga sentimental.
- **Clasificación de Intenciones (Ejercicio 6):** Al entrenar modelos para clasificar la intención de una pregunta de usuario, se observó que tanto un enfoque basado en **TF-IDF seguido de Regresión Logística** como uno que utiliza **representaciones vectoriales de Transformers con Regresión Logística** pueden alcanzar niveles de precisión comparables en el dataset específico utilizado. Dada la similitud en el rendimiento, el enfoque con TF-IDF podría ser preferible en escenarios donde la eficiencia computacional (menor tiempo de entrenamiento y predicción, y menor tamaño del modelo) sea un factor crítico.

EJERCICIO 1: Carga y Preparación Inicial de los Datos:

El proceso para disponer de la información necesaria para los análisis involucra varios pasos, realizados sin depender de herramientas específicas con nombres propios:

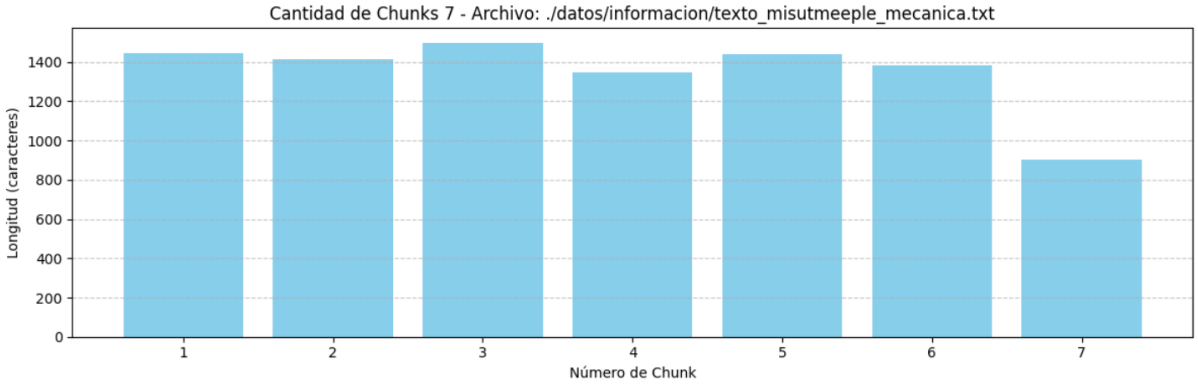
1. **Obtención de Datos Fuente:**
   Inicialmente, los datos brutos, que contienen la información textual sobre los juegos, se descargan desde Google drive. Estos datos se descargan y se almacenan localmente en el entorno de trabajo.

2. **Extracción de Archivos:**
   Los datos descargados se encuentran típicamente en comprimido. Por lo tanto, el siguiente paso es descomprimir este archivo consolidado para acceder a la colección de documentos que contiene.

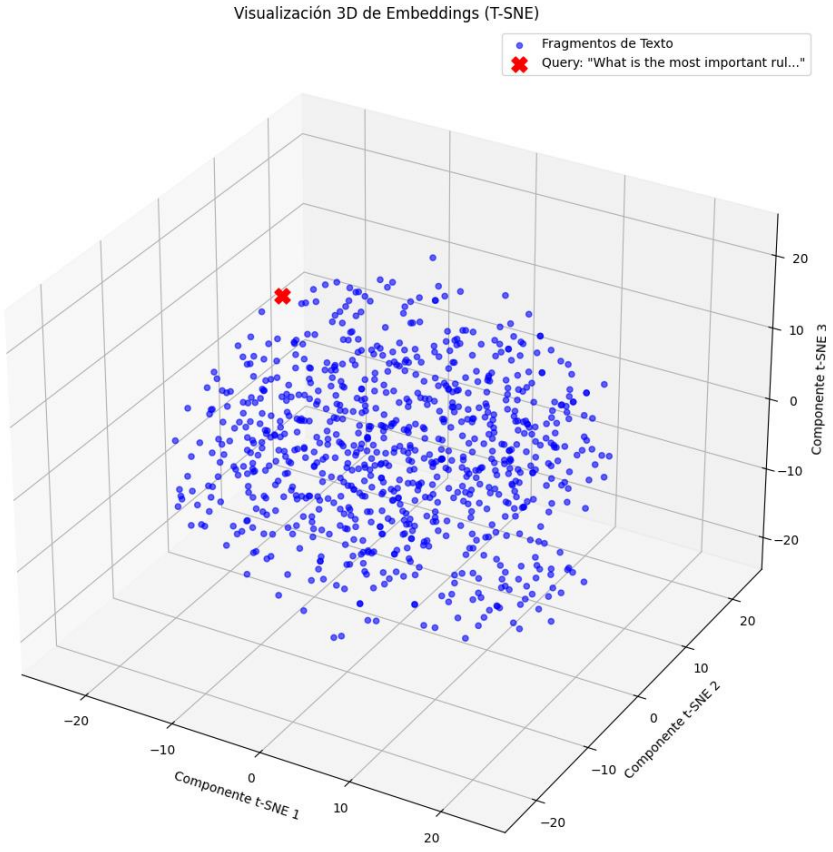
EJERCICIO 2: Fragmentación de Texto, Vectorización y Búsqueda de Similitud Semántica

El objetivo es dividir textos largos en fragmentos (chunks) más pequeños y manejables, convertirlos en representaciones vectoriales (embeddings) y luego buscar los fragmentos más relevantes para una consulta dada.

1. **Fragmentación de Texto (Chunking):**
   - Se implementa una estrategia de chunking adaptativa:
     - Textos cortos (< 2000 caracteres) se mantienen completos.
     - Para textos más largos, se intenta primero dividir por párrafos (\n\n).
     - Si la división por párrafos no es adecuada (pocos o demasiados chunks), se utiliza chonkie.SemanticChunker.
     - Se usan configuraciones de SemanticChunker diferentes para textos generales y transcripciones de video. Las transcripciones de video reciben un pre-procesamiento para añadir saltos de línea y facilitar la segmentación.
   - Se define una función eval_chunks para imprimir algunos chunks generados y visualizar la distribución de sus longitudes mediante matplotlib, ayudando a evaluar la calidad de la fragmentación.



2. **Vectorización de Chunks:**
    - Se utiliza la librería sentence-transformers con el modelo paraphrase-multilingual-MiniLM-L12-v2 para convertir cada chunk de texto en un embedding (vector numérico denso) que captura su significado semántico.
3. **Búsqueda de Similitud:**
    - Se implementa la función buscar_similares que:
        - Toma una consulta (query) del usuario.
        - Convierte la query a un embedding usando el mismo modelo.
        - Calcula la similitud entre el embedding de la query y los embeddings de todos los chunks usando tres métricas:
            - Similitud Coseno (mayor es mejor).
            - Distancia Euclidiana (menor es mejor).
            - Distancia Manhattan (menor es mejor).
        - Muestra los top_n fragmentos más similares para cada métrica.
4. **Visualización de Embeddings:**
    - La función visualizar_embeddings_3d_tsne utiliza t-SNE para reducir la dimensionalidad de los embeddings de los chunks y la query a 3D.
    - Se genera un gráfico 3D con matplotlib para visualizar la proximidad de los fragmentos a la consulta en el espacio vectorial.
    - *Conclusión observada:* La similitud coseno tiende a ofrecer los resultados más relevantes semánticamente.



EJERCICIO 3: Extracción de POS, NER y Búsqueda de Similitud basada en Sustantivos

Este ejercicio se enfoca en un análisis lingüístico más detallado y una estrategia de búsqueda de similitud diferente.

1. **Extracción de Part-of-Speech (POS) y Entidades Nombradas (NER):**
    - Se utiliza spaCy con modelos pre-entrenados para inglés (en_core_web_lg) y español (es_core_news_lg). El idioma de cada chunk se determina usando los resultados del Ejercicio 4.
    - Para cada chunk, se extraen:
        - **Sustantivos (Nouns):** Se lemmatizan para obtener su forma base.
        - **Entidades Nombradas (NER):** Como personas, organizaciones, lugares, etc.

- Esta información se almacena en un DataFrame de Pandas (df_chunks_pos_ner).

2. **Búsqueda de Similitud (Sustantivos):**
   - Se define una nueva función buscar_similares (distinta a la del Ej. 2) que:
     - Procesa la query con spaCy para extraer sus sustantivos.
     - Calcula la similitud usando dos métodos:
       - **Similitud Coseno con TF-IDF:** Se aplica TfidfVectorizer sobre los sustantivos lemmatizados de los chunks y la query, y luego se calcula la similitud coseno.

```
Top 5 - Similitud de Coseno (TF-IDF):
     chunk_id  cosine_sim                                          chunk_text
389       389    0.459742   You may even gain two of the\nsame type of to...
294       294    0.436571   However, now I am doubting if the first rule ...
464       464    0.412589  \nExcept for filling a Canteen and 1-to-1 toke...
342       342    0.411910   ie. you can only purchase the card on the top...
231       231    0.396567  There should be only 3 Parks face up available...
```

       - **Similitud de Jaccard:** Se calcula la similitud de Jaccard entre el conjunto de sustantivos de la query y el conjunto de sustantivos de cada chunk.

```
Top 5 - Similitud de Jaccard:
     chunk_id  jaccard_sim                                         chunk_text
203       203     0.333333  I recently received my Kickstarter copy and ha...
200       200     0.333333                          Base game, no expansion.
207       207     0.200000  One thing I noticed is that the Rangers are ve...
160       160     0.200000  https://www.polygon.com/22519254/trails-board-...
228       228     0.200000  And for that question, my answer is that I've ...
```

   - *Conclusión observada:* La similitud coseno con TF-IDF sobre sustantivos suele ser más robusta que Jaccard para consultas cortas.

EJERCICIO 4: Detección de Idioma

El objetivo es identificar automáticamente el idioma principal de cada documento de texto.

1. **Detección:**
   - Se utiliza la librería langdetect.

- Se crea una función detectar_idioma_en_carpeta que itera sobre los archivos, lee su contenido y usa langdetect.detect() para predecir el idioma.
- Los resultados (nombre del archivo e idioma detectado) se guardan en un DataFrame (df_idioma_archivos).
- Esta información es crucial para el Ejercicio 3, para seleccionar el modelo de spaCy correcto.

	archivo	idioma
0	htp_videos_es_6.txt	es
1	precios_parks.txt	en
2	texto_foros_strategy.txt	en
3	htp_videos_es_7.txt	es
4	texto_misutmeeple_contenido.txt	es
5	texto_foros_variants.txt	en
6	htp_videos_es_5.txt	es
7	reviews_videos_en_3.txt	en
8	reviews_videos_en_2.txt	en
9	htp_videos_es_4.txt	es
10	htp_videos_es_1.txt	es
11	reviews_videos_en_1.txt	en
12	reviews_videos_es_4.txt	es
13	reglas_pdf_espanol.txt	es

EJERCICIO 5: Análisis de Sentimientos en Reseñas

Este ejercicio se centra en determinar el sentimiento expresado en reseñas de juegos y buscar reseñas con sentimientos similares.

1. **Carga de Reseñas:**
   - Se carga un archivo CSV (./datos/estadisticas/reseñas_parks.csv) que contiene reseñas textuales.
2. **Análisis de Sentimientos:**
   - Se utiliza un pipeline de la librería transformers con el modelo pre-entrenado nlptown/bert-base-multilingual-uncased-sentiment.
   - Este modelo predice una calificación en estrellas para cada reseña.
   - La calificación en estrellas se mapea a una etiqueta de sentimiento (positivo, neutro, negativo).

- Las puntuaciones de estrellas y las etiquetas de sentimiento se añaden como nuevas columnas al DataFrame de reseñas.

	puntaje	reseña	score	sentimiento
0	5.0	I can often feel upon finishing a rulebook whe...	2 stars	negativo
1	8.0	I have the original version with the original,...	4 stars	positivo
2	7.0	I really like this game. I think it's got enou...	4 stars	positivo
3	7.3	I love the National Parks. The art in this gam...	4 stars	positivo
4	8.0	Both expansions, promo card and upgraded piece...	5 stars	positivo

3. **Búsqueda de Reseñas Similares por Sentimiento:**
    - Se implementa la función buscar_reseñas_similares:
        - Analiza el sentimiento de la query del usuario usando el mismo pipeline.
        - Filtra el DataFrame de reseñas para incluir solo aquellas con el mismo sentimiento que la query.
        - Vectoriza la query y las reseñas filtradas usando el modelo sentence-transformers (el mismo que en Ej. 2).
        - Calcula la similitud coseno para encontrar las reseñas más parecidas dentro de la categoría de sentimiento identificada.

EJERCICIO 6: Clasificación de Intenciones de Consultas

El objetivo es entrenar un modelo capaz de clasificar una pregunta de usuario en una de tres intenciones predefinidas: información, relaciones o estadística.

1. **Dataset:**
    - Se define un dataset manualmente etiquetado, donde cada entrada es una tupla (etiqueta_numérica, texto_pregunta).
2. **Entrenamiento y Evaluación de Modelos:**
    - El dataset se divide en conjuntos de entrenamiento y prueba.
    - Se exploran dos enfoques para la clasificación:

- **Enfoque 1: TF-IDF + Regresión Logística:**
  - Las preguntas se convierten en vectores TF-IDF, eliminando stopwords en español e inglés.
  - Se entrena un clasificador de LogisticRegression sobre estos vectores.

```
Precisión Regresión Logística: 0.9625
Reporte de clasificación Regresión Logística:
              precision    recall  f1-score   support

           0       0.98      0.95      0.96        43
           1       0.91      0.95      0.93        21
           2       1.00      1.00      1.00        16

    accuracy                           0.96        80
   macro avg       0.96      0.97      0.96        80
weighted avg       0.96      0.96      0.96        80
```

- **Enfoque 2: Embeddings de Transformer + Regresión Logística:**
  - Las preguntas se convierten en embeddings usando sentence-transformers (modelo sentence-transformers/all-mpnet-base-v2).
  - Se entrena un clasificador de LogisticRegression sobre estos embeddings.

```
Precisión Regresión Logística: 0.9625
Reporte de clasificación Regresión Logística:
              precision    recall  f1-score   support

           0       0.98      0.95      0.96        43
           1       0.91      0.95      0.93        21
           2       1.00      1.00      1.00        16

    accuracy                           0.96        80
   macro avg       0.96      0.97      0.96        80
weighted avg       0.96      0.96      0.96        80
```

- Ambos modelos se evalúan usando métricas como precisión (accuracy) y el reporte de clasificación (precision, recall, F1-score por clase).
- *Conclusión observada:* Para este dataset específico, ambos enfoques logran una precisión similar, sugiriendo que el modelo TF-IDF + Regresión Logística podría ser preferible debido a su menor costo computacional.

Conclusión

Los resultados obtenidos en cada ejercicio demuestran la capacidad de estas técnicas para extraer información valiosa y generar insights a partir de datos textuales no estructurados.

En la **búsqueda semántica (Ejercicio 2)**, utilizando representaciones vectoriales densas de los fragmentos de texto, se determinó que la **similitud coseno** tiende a ofrecer los resultados más relevantes semánticamente. Esta métrica superó a la distancia Euclidiana y Manhattan al capturar mejor el significado contextual de las consultas, recuperando fragmentos de diversas fuentes (textos de videos, foros, guías) que se alineaban con la intención de la búsqueda, mientras que las otras métricas a menudo se limitaban a coincidencias superficiales, como menciones en transcripciones de video.

Para la **búsqueda basada en sustantivos (Ejercicio 3)**, se contrastó la similitud coseno con TF-IDF frente a la similitud de Jaccard. Se concluyó que la **similitud coseno con TF-IDF** aplicada sobre los sustantivos lemmatizados devuelve mejores resultados. Esto se debe a que es menos dependiente de la cantidad absoluta de elementos coincidentes y se enfoca más en la distribución y peso de los términos, lo cual es ventajoso cuando se comparan consultas (generalmente cortas) con fragmentos de corpus más extensos.

La **detección automática de idioma (Ejercicio 4)** fue un paso instrumental, permitiendo la correcta aplicación de herramientas lingüísticas específicas para cada lengua en etapas posteriores, como la selección de los modelos de spaCy adecuados para el análisis morfosintáctico y de entidades en el Ejercicio 3.

El **análisis de sentimientos (Ejercicio 5)**, empleando modelos pre-entrenados de la librería transformers, clasificó eficazmente la polaridad de las reseñas. Al combinar esta clasificación con la búsqueda de similitud semántica, se pudo refinar la recuperación de información, no solo encontrando reseñas temáticamente similares a una consulta, sino también aquellas que compartían la misma carga sentimental.

Finalmente, en la **clasificación de intenciones de consultas (Ejercicio 6)**, se evaluaron dos enfoques: TF-IDF con Regresión Logística y embeddings de Transformer con Regresión Logística. Se observó que ambos modelos alcanzaron una precisión idéntica en el dataset proporcionado. Dada esta paridad en el rendimiento, el modelo de **Regresión Logística con vectorización TF-IDF** se perfila como la mejor opción para este caso específico, principalmente por ser significativamente más liviano y rápido en términos computacionales que el modelo basado en Transformers.

Referencias

- **Chonkie (Semantic Text Splitter):**
    - Documentación oficial: https://www.chonkie.ai/
- **Sentence-Transformers:**
    - Reimers, N., & Gurevych, I. (2019). *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. En Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing.
    - Sitio web oficial: https://www.sbert.net
- **spaCy:**
    - Honnibal, M., Montani, I., Van Landeghem, S., & Boyd, A. (2020). *spaCy: Industrial-strength Natural Language Processing in Python*.
    - Sitio web oficial: https://spacy.io
- **Langdetect:**
    - Repositorio de GitHub: https://github.com/fedelopez77/langdetect
- **Hugging Face Transformers:**
    - Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Brew, J. (2020). *Transformers: State-of-the-Art Natural Language Processing*. En Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations.
    - Sitio web oficial: https://huggingface.co/transformers
- **Scikit-learn:**
    - Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825-2830.
    - Sitio web oficial: https://scikit-learn.org/
- **NLTK (Natural Language Toolkit):**
    - Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media Inc.
    - Sitio web oficial: https://www.nltk.org/