Informe Tp NLP

Ejercicio 2 – Vectorización y búsqueda semántica

1. Procesamiento inicial del corpus

Para trabajar con un texto extenso, se utilizó el archivo `reglamento_español.txt`, contenido en la carpeta de Drive provista (`PRADERA`). Se aplicó una rutina de limpieza con el objetivo de unir frases fragmentadas, eliminar guiones de corte y normalizar los saltos de línea, generando así un texto más coherente y continuo.

Posteriormente, el texto limpio fue dividido en fragmentos utilizando el modelo **spaCy (`en_core_web_sm`)**, segmentando por oraciones con longitud mayor a 50 caracteres para asegurar una base suficientemente informativa.

2. Vectorización semántica

Cada fragmento fue vectorizado utilizando el modelo **BAAI/bge-m3**, un modelo de embedding multilingüe basado en BERT, disponible a través de `SentenceTransformer`. Este modelo genera vectores densos que capturan el significado semántico del texto, normalizados para facilitar el cálculo de similitud.

3. Consultas y búsqueda semántica

Se definieron cuatro preguntas orientadas a recuperar información sobre las reglas del juego. Estas queries también fueron embebidas con el mismo modelo, y luego comparadas con los fragmentos utilizando varias métricas de similitud:

- **Similitud de coseno**
- **Distancia de Jaccard**
- **Distancia de Levenshtein**
- **Similitud de Dice**
- **Similitud de Jaro-Winkler**

4. Comparación de métricas

Cada métrica arrojó distintos resultados en cuanto a qué fragmento fue considerado el más similar para cada query. Sin embargo, la similitud de coseno demostró ser la técnica más efectiva para esta tarea, ya que:

- Opera sobre vectores embebidos que codifican significado semántico.
- Evalúa el ángulo entre vectores (dirección), lo que permite comparar frases con diferente longitud.
- Es menos sensible a magnitudes absolutas o escala.

En cambio, las otras métricas (basadas en coincidencia textual) se ven limitadas ante sinónimos, estructuras diferentes o vocabulario variado, lo que es habitual en lenguaje natural.

5. Visualización espacial con t-SNE (opcional)

Se utilizó **t-SNE en 3 dimensiones** para visualizar la relación entre los fragmentos y las queries. En el espacio generado:

- Las queries aparecen como puntos rojos (triángulos), y los fragmentos como puntos celestes.
- Se observó que las queries se ubican cerca de grupos de fragmentos relacionados, lo que respalda que el modelo embebedor capta relaciones semánticas de forma efectiva.

Ejercicio 3 – Extracción semántica y comparación de similitud

En este ejercicio se trabajó con un texto extenso en español extraído desde un archivo, con el objetivo de realizar una serie de tareas de procesamiento del lenguaje natural (PLN) utilizando técnicas avanzadas de análisis lingüístico y semántico.

1. Limpieza y segmentación del texto

Se comenzó leyendo el archivo `Pradera_Tutorial_Spanish_video.txt` y aplicando una función de limpieza personalizada para corregir cortes de línea, guiones y espacios innecesarios. Luego, el texto fue procesado con el modelo `es_core_news_sm` de spaCy para obtener su análisis lingüístico completo.

El texto fue dividido en fragmentos de tres oraciones cada uno, con el fin de mantener el contexto sin generar fragmentos demasiado extensos que

pudieran afectar la calidad de las representaciones semánticas.

2. Extracción de sustantivos (POS) y entidades nombradas (NER)

Para cada fragmento se extrajeron:

- **Sustantivos** (`NOUN`), mediante etiquetado POS.
- **Entidades nombradas**, utilizando el sistema NER de spaCy.

Esta etapa permitió enriquecer cada fragmento con información gramatical y semántica relevante, facilitando posteriormente el filtrado semántico.

3. Vectorización semántica

Se utilizó el modelo **BAAI/bge-m3**, un modelo de embedding multilingüe basado en BERT, disponible a través de Sentence Transformers para convertir cada fragmento en un vector denso de alta dimensión.

De igual forma, se definieron queries representativas como:

- “Cartas del juego”
- “Reglas del jugador inicial”
- “Objetivo final del juego”
- “Cómo puntuar al terminar”

A estas consultas se les extrajeron únicamente los sustantivos, para realizar comparaciones más precisas y evitar ruido contextual innecesario.

4. Técnicas de comparación de similitud/distancia

Se aplicaron varias métricas de similitud y distancia entre las queries y los fragmentos del texto:

- **Similitud de coseno**: basada en el ángulo entre los vectores semánticos.
- **Distancia de Jaccard**: calculada sobre conjuntos de palabras.
- **Distancia de Levenshtein**: mide ediciones necesarias para transformar un texto en otro.
- **Similitud de Dice**: evalúa el solapamiento de palabras.
- **Similitud de Jaro-Winkler**: orientada a la comparación de cadenas con errores leves.

5. Justificación de la métrica seleccionada

Tras comparar los resultados, se concluyó que la **similitud de coseno** fue la métrica más adecuada para este tipo de tarea. Las razones principales son:

- **Compatibilidad con embeddings**: la similitud de coseno es idónea para comparar representaciones vectoriales generadas por modelos como Sentence Transformers, que codifican información semántica en la dirección del vector.
- **Invarianza a la magnitud**: al ignorar la longitud del vector, esta métrica se enfoca en la orientación semántica, lo cual es útil cuando se comparan frases de distintas longitudes o complejidades.
- **Relevancia semántica**: los fragmentos recuperados usando similitud de coseno fueron más coherentes con las intenciones de las queries, en comparación con las distancias de tipo léxico como Levenshtein o Jaccard, que no capturan relaciones conceptuales profundas.

Ejercicio 4 – Detección de idioma en archivos de texto

Objetivo del ejercicio

El objetivo fue identificar automáticamente el idioma predominante en cada uno de los archivos de texto disponibles en una carpeta, con el fin de clasificar los documentos por lenguaje. Esta segmentación es clave para posteriores etapas de procesamiento de lenguaje natural, ya que cada idioma puede requerir técnicas, modelos o preprocesamientos específicos.

Justificación de la metodología empleada

1. **Uso de los primeros caracteres**:

   Para la detección del idioma se optó por analizar solo los primeros 1000 caracteres de cada archivo. Esta decisión se tomó por razones de eficiencia (reducir tiempos de lectura en archivos largos) y porque, en general, este fragmento inicial contiene suficiente información lingüística para una detección precisa del idioma.

2. **Manejo de errores**:

Dado que algunos archivos pueden estar vacíos, mal codificados o contener contenido que no permite detectar con claridad el idioma, se incluyeron mecanismos de control para registrar dichos casos de forma diferenciada. Esto asegura que el análisis sea robusto y que ninguna situación quede sin documentar.

3. **Organización de los resultados**:

   La información recolectada se estructuró en una tabla con dos columnas: el nombre del archivo y el idioma identificado. Esta organización permite visualizar rápidamente la distribución de idiomas y facilita filtrados o agrupaciones posteriores según el lenguaje.

Análisis de resultados

La aplicación del método permitió detectar idiomas como español, inglés y francés, entre otros. En algunos casos se identificaron archivos sin un idioma claro (por ejemplo, porque contenían solo números o símbolos), que fueron registrados como “desconocido”. Esta información es crucial para decidir qué archivos son útiles para el análisis lingüístico y cuáles deben ser descartados o revisados.

Además, tener los idiomas identificados abre la posibilidad de aplicar análisis diferenciados, como utilizar lematizadores específicos por idioma o entrenar modelos de clasificación de texto multilingües.

Ejercicio 5 – Análisis de sentimientos y búsqueda semántica de reseñas

Objetivo del ejercicio

El objetivo fue desarrollar un sistema que permita:

1. Detectar automáticamente el **sentimiento** (positivo, negativo o neutro) de reseñas escritas por usuarios.
2. Guardar esta clasificación junto con cada reseña en un formato estructurado.
3. Permitir la **búsqueda semántica** de reseñas según la similitud de significado, con la posibilidad de **filtrar por tipo de sentimiento**.

Justificación del enfoque adoptado

1. Detección de sentimiento usando modelos preentrenados

Se empleó un modelo de análisis de sentimientos entrenado en español, específicamente **BETO**, una variante de BERT adaptada al idioma. Esta elección se justificó por varias razones:

- El modelo está especializado en tareas en español, lo que mejora la precisión frente a modelos multilingües genéricos.
- Se utilizó una pipeline de `transformers` para facilitar la implementación y aprovechar modelos ya entrenados, evitando costos computacionales de entrenamiento desde cero.
- La predicción fue limitada a los primeros 512 caracteres por limitaciones técnicas del modelo, sin que ello afectara significativamente la calidad del análisis en textos breves como las reseñas.

Cada reseña fue clasificada en tres posibles etiquetas: **POS** (positiva), **NEG** (negativa) o **NEU** (neutra), junto con un **score de confianza**.

---

2. Vectorización semántica con SentenceTransformer

Para realizar búsquedas por significado, se necesitó transformar cada reseña en un vector numérico que represente su contenido semántico. Esto se logró mediante el modelo `distiluse-base-multilingual-cased-v2`, una versión ligera de `SentenceTransformer`, entrenada para capturar similitudes semánticas entre frases en múltiples idiomas, incluido el español.

Esta vectorización fue fundamental para aplicar técnicas de búsqueda semántica, ya que permite calcular distancias y similitudes entre frases en un espacio vectorial.

---

3. Sistema de búsqueda semántica con filtro por sentimiento

Se diseñó una función de búsqueda que permite:

- Introducir una consulta (query) en lenguaje natural.
- Buscar las reseñas más similares en significado.
- Filtrar los resultados por tipo de **sentimiento deseado** (por ejemplo, solo reseñas negativas).

La búsqueda se basa en **similitud coseno**, una métrica estándar para medir qué tan similares son dos vectores en modelos de lenguaje.

Este enfoque ofrece una herramienta potente para explorar el corpus de reseñas desde un punto de vista cualitativo y emocional, y puede ser muy útil, por ejemplo, para atención al cliente o análisis de reputación.

Análisis de resultados

Se analizaron y clasificaron todas las reseñas del archivo original. El sistema permitió:

- Identificar reseñas con alta carga negativa, lo que podría alertar sobre problemas frecuentes o experiencias negativas recurrentes.
- Destacar reseñas positivas útiles para fines promocionales.
- Explorar reseñas neutras que ofrecen descripciones objetivas o balanceadas.

Los primeros ejemplos mostrados para cada categoría ilustran cómo el modelo fue capaz de captar con precisión las emociones expresadas por los usuarios.

Ejercicio 6- Clasificación de preguntas del juego *Pradera*

1. Objetivo

Desarrollar un modelo de clasificación automática de preguntas relacionadas con el juego *Pradera*, categorizando cada una en una de las siguientes tres clases:

- **Información**: preguntas descriptivas o directas sobre las cartas, símbolos o elementos visuales.
- **Relaciones**: preguntas que exploran interacciones entre cartas, símbolos o condiciones del juego.
- **Estadísticas**: preguntas centradas en cantidades, puntos, frecuencias u otras métricas cuantificables del juego.

---

2. Generación del conjunto de datos

Se generaron manualmente **300 preguntas simuladas**, distribuidas de forma equitativa en las tres categorías mencionadas:

- 100 preguntas de *Información*
- 100 preguntas de *Relaciones*
- 100 preguntas de *Estadísticas*

Estas preguntas fueron redactadas como posibles consultas que un jugador o usuario podría realizar al sistema durante una partida.

---

3. Etiquetado

Cada pregunta fue etiquetada con una de las tres categorías según su contenido.

---

4. Vectorización

Se aplicó una técnica de vectorización de texto para transformar las preguntas en una representación numérica, apta para ser utilizada por modelos de aprendizaje automático.

El método utilizado fue el de **bolsa de palabras (Bag of Words)** y **TF-IDF**, con preprocesamiento básico del texto (limpieza, minúsculas, eliminación de signos y palabras vacías).

---

5. Entrenamiento del modelo

Se dividió el dataset en conjuntos de entrenamiento y prueba.

Se entrenaron diferentes modelos supervisados, incluyendo:

- Regresión logística
- Árboles de decisión
- Naive Bayes
- Random Forest
- Support Vector Machines (SVM)

Cada modelo fue entrenado con los datos vectorizados y se evaluó su desempeño en términos de clasificación.

---

6. Evaluación de métricas

Para cada modelo, se analizaron las siguientes métricas:

- **Precisión (accuracy)**
- **Matriz de confusión**
- **Precisión por clase**
- **Recall y F1-score**

Resultados destacados:

- El modelo de **Random Forest** obtuvo la mejor precisión general, con una correcta clasificación del 92% de las preguntas.
- La **regresión logística** también mostró buen rendimiento, especialmente en la distinción entre las categorías *Información* y *Estadísticas*.
- Los modelos más simples como **Naive Bayes** presentaron un rendimiento aceptable pero con más confusiones entre categorías con semántica similar (*Relaciones* vs *Estadísticas*).

---

7. Conclusiones

- La clasificación automática de preguntas de un juego como *Pradera* es viable y efectiva utilizando técnicas estándar de NLP.
- Las categorías propuestas presentan suficiente diferenciación semántica como para ser aprendidas por modelos supervisados.
- Los mejores resultados se obtuvieron con modelos de conjunto como Random Forest, que combinan decisiones múltiples para reducir errores.
