Trabajo Práctico N°1

Materia: **IA4.2 Procesamiento Del Lenguaje Natural**
Carrera: **Tecnicatura Universitaria en Inteligencia Artificial**

Primer integrante: **Nicolás Noir: N-1273/4**
Segundo integrante: **Ignacio Eloy González: G-5933/1**

Primer profesor: **Juan Pablo Manson**
Segundo profesor: **Alan Geary**

Fecha: 06/11/2024

Universidad Nacional de Rosario
Facultad de Ciencias Exactas, Ingeniería y Agrimensura

ÍNDICE

INTRODUCCIÓN

Los objetivos y problemas del trabajo práctico se desarrollan dentro de una situación hipotética en la que una persona iba a ir de viaje un mes a la playa pero iba a tener cuatro días de lluvia, imposibilitando poder disfrutar del exterior. Para esta situación, se debía desarrollar primero un algoritmo que según el sentimiento que comunicara esta persona, se debía determinar su estado de ánimo mediante un modelo de clasificación de sentimientos, que debía ser entrenado previamente con diferentes frases para los tres posibles estados de ánimo: triste, neutral y alegre. Luego de entrenar el modelo utilizando una regresión logística, se calificaba este modelo para así tener noción de que tan bien o mal entrenado estaba este modelo

Seguidamente, debía realizarse otro modelo que satisficiera mediante la devolución de libros, películas o juegos de mesa las preferencias que el usuario ingresa en conjunto con su estado de ánimo en un menú interactivo. El programa deberá comprobar la frase con diversas estructuras de texto provenientes de un modelo pre-entrenado. Posteriormente, el programa enseñará los títulos de los tópicos más relacionados a la preferencia anteriormente escrita, según se elija.

RESUMEN

En este trabajo práctico vamos a entrenar un modelo de clasificación para el análisis de sentimiento de una frase en el contexto dado. En este clasificador utilizaremos regresión logística como modelo a entrenar, para las variables explicativas utilizaremos un embedding de las frases generadas por Chat GPT y para la variable target utilizaremos las clases felíz, neutral y triste (codificadas en 0, 1 y 2).

Antes de hacer la recomendación fue necesario crear un dataset para los libros mediante la extracción de la información con web scraping. También fue necesario limpiar los datasets de las películas y juegos de mesa.

Para la recomendación primero vamos a hacer reconocimiento de entidades nombradas, así podemos segmentar la búsqueda. Después dependiendo del resultado de ese reconocimiento poder hacer una búsqueda semántica, mediante la similitud del coseno, más precisa cambiando la información pasada al embedding de los libros, películas y juegos de mesa.

Luego, toda esta información es plasmada utilizando la librería IpyWidgets, mediante un menú interactivo para poder ingresar, los sentimientos y las preferencias. Luego dejando la opción a seleccionar si desea visualizar películas, libros, juegos de mesa o todos a la vez, pudiendo variar la cantidad de cada uno que quiera ver.

METODOLOGÍA

Para el trabajo, se implementaron las siguientes fuentes:

- Dataset juegos de mesa: https://drive.google.com/file/d/1yIWOgUV5WyskQvmq48QvF2Lzr0LxpAdq/view?usp=drive_link
- Dataset películas: https://drive.google.com/file/d/1yIWOgUV5WyskQvmq48QvF2Lzr0LxpAdq/view
- Página para la extracción del dataset de los libros: Top 1000 | Project Gutenberg

Las librerías que se utilizaron son:

- **Pandas**: Para poder trabajar con los datasets de manera óptima y cómoda
- **Files**: Para poder descargar los diferentes dataset que fueron modificados
- **SentenceTransformer**: Para poder importar un modelo transformer pre entrenado y hacer embeddings.
- **Util**: Proporciona funciones que facilitan diferentes cálculos como la similitud del coseno y la búsqueda semántica entre embeddings
- **re**: Para poder hacer uso de las regex proporcionadas por python
- **requests**: Para poder hacer peticiones a las diferentes páginas para el uso del web scraping
- **BeautifulSoup**: Para poder extraer la información de una página web, es decir, hacer web scraping.
- **Time**: Para poder dejar un tiempo entre cada solicitud para que no se detecte un posible robo de información dentro de la página.
- **train_test_split**: Para poder dividir los datos entre train y test
- **LogisticRegression**: Para poder entrenar el modelo de sentimientos
- **accuracy_score y classification_report**: Para poder evaluar el modelo de regresión logística

- **GoogleTranslator**: Para traducir las preferencias y mejorar la recomendación al comparar todo en un único idioma.
- **Gliner**: Para el reconocimiento de entidades nombradas.
- **Ipywidgets**: Para implementar una interfaz sencilla utilizando las siguientes clases de esta librería: Text, Button, Output, VBox, IntSlider, HBox.
- **Torch**: Para la transformación del tipo de dato de los embeddings.
- **Warnings**: Para ignorar advertencias.

El software que se utilizó es:

- **Google Colab** como “IDE”.
- **GitHub** como controlador de versiones.
- **ChatGPT** para la creación del dataset de entrenamiento del modelo de análisis de sentimiento

CÓDIGO DATASET:

1. Se realizó el web scraping mediante la utilización de las librerías BeautifulSoup. requests & time. La primera parte se busca encontrar el link de cada libro dentro de la página de gutenberg, luego con ese mismo link, se busca guardar la información de cada libro contenido, guardando el autor, el título, la descripción, y los diferentes géneros en una lista para facilitar el posterior análisis. Cada una de estas búsquedas individuales de libros, está separada por un segundo para obviar cualquier tipo de problema con la página.

2. Se eliminan las columnas innecesarias de los datasets de juegos de mesa y de películas, para libros no es un trabajo necesario ya que al momento del web scraping estos factores se tuvieron en cuenta. Luego se renombran las columnas para facilitar la creación de embedding y se limpia las fechas de vida de los autores de los libros, ya que esto genera error al momento de realizar el encoding. Esto para reducir el tamaño de los datasets y facilitar el manejo del mismo.

3. Al final se genera un dataset de entrenamiento para el modelo de detección de sentimientos, proporcionado por Chat Gpt, generando 90 frases para cada tipo de sentimiento (triste, neutral y alegre) centrando la temática de las frases en el clima y sentimientos banales.

CÓDIGO MODELOS:

1. Se realiza el entrenamiento del modelo de clasificación de sentimientos utilizando el dataset anteriormente creado por ChatGPT, el modelo elegido para la creación de embeddings es paraphrase-multilingual-MiniLM-L12-v2 ya que este modelo fue el que mejores resultados dio. Se probaron diferentes modelos para el embedding y luego de un entrenamiento utilizando una regresión logística, evaluando en precisión, recall y f1, este fue el que mejores resultados otorgó. También se probó el modelo de all mpnet base v2, pero aunque este modelo cuente mayor espacio vectorial y sea más popular, para el modelo de sentimientos, devolvió resultados menos precisos. También el modelo all MiniLM L6 v2 y enseñó valores incluso inferiores.

Reportes de métricas de todos los modelos:

- sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2. (Figura 1.1)

```
Precisión Regresión Logística: 0.9259259259259259
Reporte de clasificación Regresión Logística:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        17
           1       0.84      0.94      0.89        17
           2       0.94      0.85      0.89        20

    accuracy                           0.93        54
   macro avg       0.93      0.93      0.93        54
weighted avg       0.93      0.93      0.93        54
```

Figura 1.1.

- all-mpnet-base-v2. (Figura 1.2)

```
Precisión Regresión Logística: 0.7592592592592593
Reporte de clasificación Regresión Logística:
              precision    recall  f1-score   support

           0       0.79      0.88      0.83        17
           1       0.71      0.88      0.79        17
           2       0.79      0.55      0.65        20

    accuracy                           0.76        54
   macro avg       0.76      0.77      0.76        54
weighted avg       0.76      0.76      0.75        54
```

Figura 1.2

- sentence-transformers/all-MiniLM-L6-v2. (Figura 1.3)

```
Precisión Regresión Logística: 0.6296296296296297
Reporte de clasificación Regresión Logística:
              precision    recall  f1-score   support

           0       0.52      0.65      0.58        17
           1       0.60      0.71      0.65        17
           2       0.85      0.55      0.67        20

    accuracy                           0.63        54
   macro avg       0.66      0.63      0.63        54
weighted avg       0.67      0.63      0.63        54
```

Figura 1.3

Luego se predicen los sentimientos ingresados por el usuario, clasificándolos en tres clases distintas: triste, neutral y feliz.

2. Se hace uso del mismo modelo anteriormente usado en el embedding de las frases del clasificador de sentimientos, con el fin de obtener el embedding de la preferencia del usuario ingresada dentro de un menú interactivo creado con Ipywidget. La preferencia del usuario se traduce al inglés para una mejor evaluación dentro de los tópicos, pudiendo sugerir temas como géneros, actores, directores, escritores y descripciones de cualquiera de estos argumentos. Esto es gracias a NER (Gliner) que nos permite encontrar directores de cine, actores de cine y autores de libros. En un momento se pensó en utilizar más etiquetas de reconocimiento en NER, siendo la palabra película, libro y juego de mesa, para delimitar la búsqueda semántica a una de las tres, esta función tenía la contraparte de que si el usuario negaba esta sugerencia, el modelo no tiene la capacidad de comprender esta negación. Por lo cual fue eliminada del código. También se necesitó de dos embeddings diferentes para la búsqueda semántica, uno para cuando no se detectan autores, directores o actores y el otro para cuando si se detectan. En el primer caso es el embedding de la concatenación de las categorías y la descripción de cada libro, película o juego de mesa. En el otro caso si se detecta un autor, la búsqueda se limita a libros y el embedding es hecho con el autor del libro únicamente y en caso de que se detecte un director o actor, el embedding es hecho con la concatenación del director y los actores. Figura 2.1

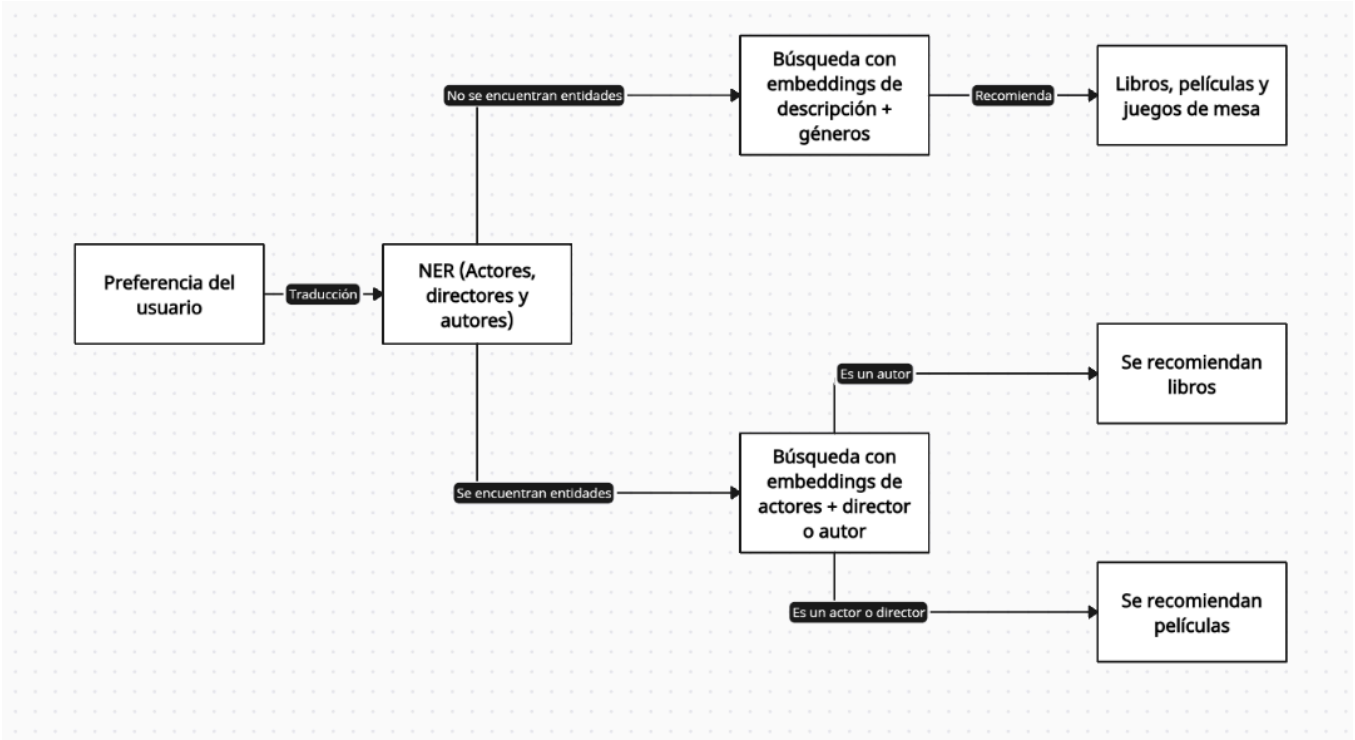


Figura 2.1

Se decidió utilizar la preferencia de la persona para el embedding para la búsqueda semántica, ya que se probó incluir la clasificación, el prompt completo y ambas cosas llegando a un resultado peor, tomando como métrica la similitud del coseno entre los embeddings.

3. Ambos modelos, sentimientos y preferencias están operativos dentro del menú, donde se podrá completar con el estado de ánimo del usuario y su preferencia, para posteriormente ser analizada y satisfacida. Luego se dejarán a disposición del consumidor botones donde podrá elegir alguno de los tres argumentos y además se podrá elegir visualizar ambos temas a la vez. También se podrá preferir la cantidad de disyuntivas a apreciar por medio de una barra dinámica que varía de cero a cinco

RESULTADOS

- Se ingresa:
  - me siento realmente esperanzado
  - quiero una historia de aventura y fantasía

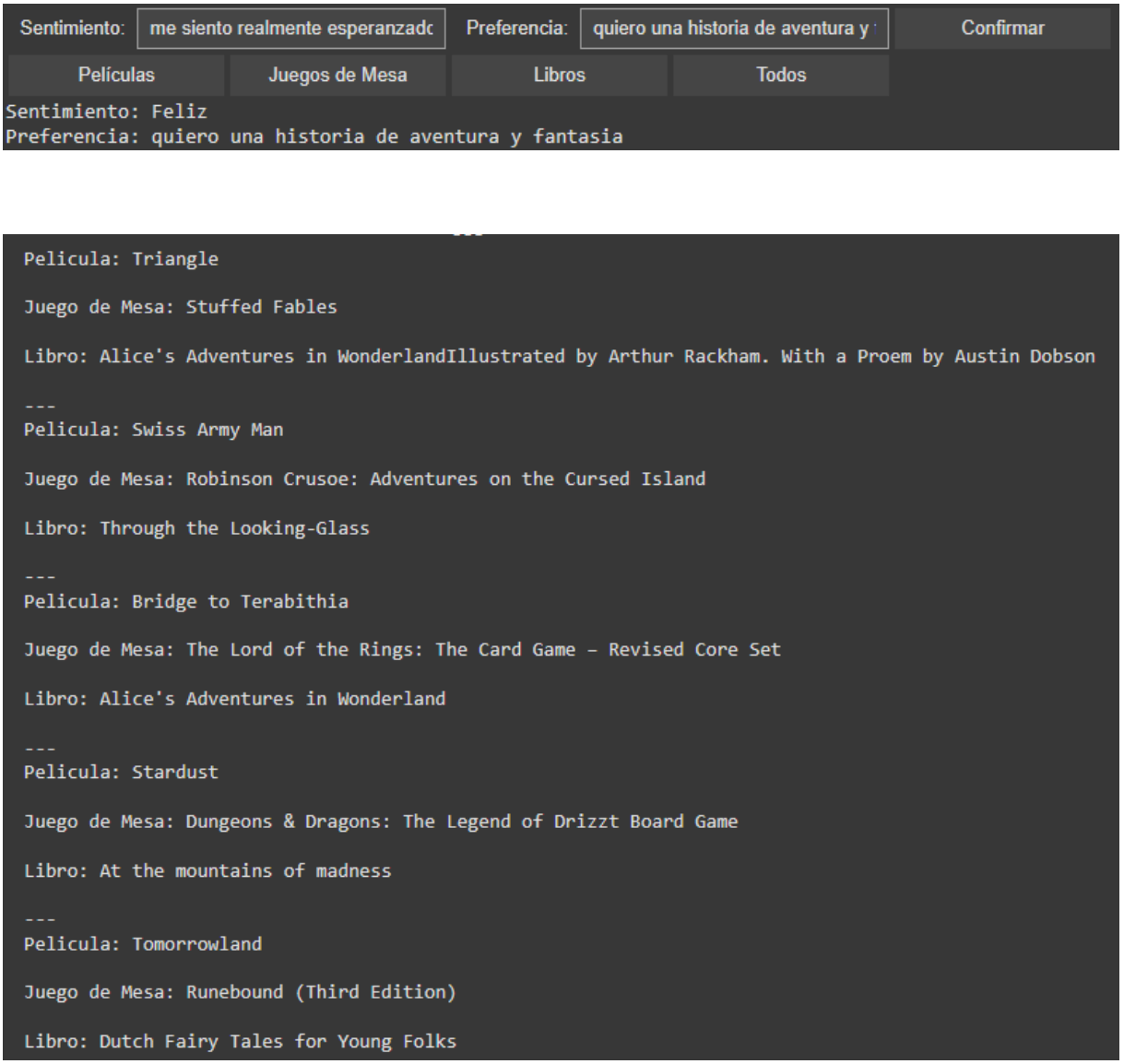


Figura 3.1

Figura 3.1 Dentro de los temas de libros y juegos de mesa, el modelo resulta ser altamente satisfactorio para esta temática, mientras que para apeliculas es un poco inferior a estos, ya que aunque este devuelva películas que tienen como género aventura o fantasía, las primeras carecen de ambos géneros, pudiendo este modelo devolver películas como el Hobbit o Harry Potter que satisfacen estos géneros

- Se ingresa:
  - El dia esta muy lluvioso y arruinó mid días
  - Quiero una pelicula donde actua Leonardo Dicaprio

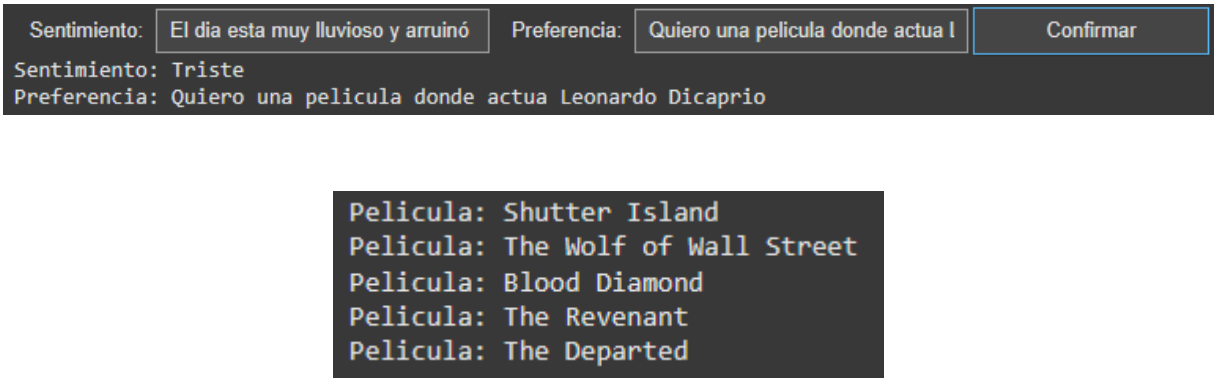


Figura 3.2

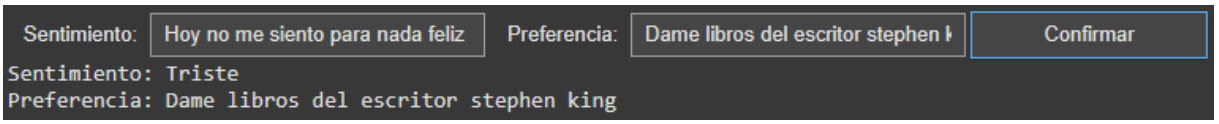
Figura 3.2 En este caso el modelo de sentimientos volvió a dar muy buenos resultados mientras que cuando se buscan actores como en este caso Leonardo Dicaprio, el modelo resulta ser muy eficaz, interpretando que queremos películas donde acute esta persona

- Se ingresa:
  - No me siento ni triste ni alegre
  - Recomendame un libro escrito por Aristoteles

Figura 3.3

Figura 3.3 El modelo de sentimientos resulta ser muy bueno en reiteradas oportunidades y en diferentes temáticas. También es el caso donde buscamos libros de Aristoteles, donde no solo encuentra libros escritos por el histórico personaje, sino que también logra devolver dos libros Platón, estando estos dos personajes altamente relacionados, Siendo Platón maestro de Aristoteles y compartiendo ideas sobre la concepción de la forma (eîdos) como causa del ser y del conocimiento de las cosas

- Se ingresa:
  - Hoy no me siento para nada feliz
  - Dame libros del escritor stephen king



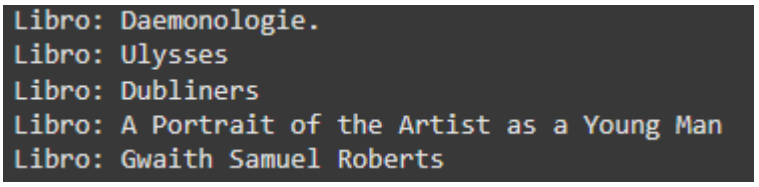
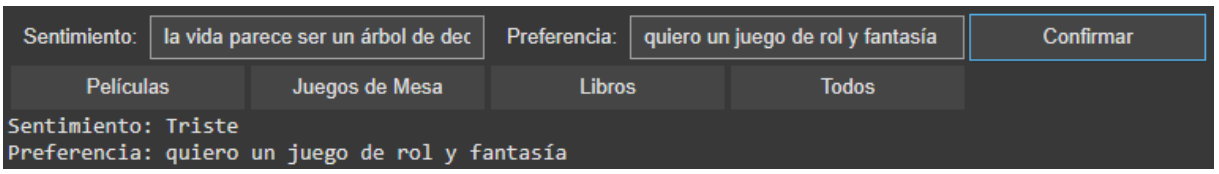


Figura 3.4

Figura 3.4 En términos del modelo de sentimientos, este es grandiosamente preciso en cuanto a sus evaluaciones. En este caso en que pedimos libros de Stephen King, el modelo no es solo que no reconoce al autor, sino que confunde a este, con el novelista James Joyce en reiteradas ocasiones. También puede ser que en el dataset no se encuentren libros de Stephen King y le sea imposible al modelo traer libros del mismo.

- Se ingresa:
  - la vida parece ser un árbol de decisiones, donde siempre elijo la opción incorrecta
  - quiero un juego de rol y fantasía



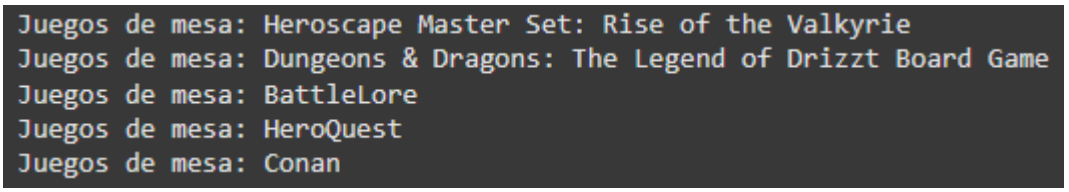


Figura 3.5

Figura 3.5 Es gratamente sorprendente los resultados del modelo de sentimientos, aunque la frase dada pueda ser confusa, este la evalúa de forma correcta. En parte de los juegos de mesa, también se obtienen muy buenos resultados, todos siguiendo los géneros que se le fueron asignados al modelo

CONCLUSIONES

En este trabajo se tuvieron buenos resultados, primero en el modelo clasificador del estado de ánimo, donde con un dataset pequeño se pudo entrenar un modelo que, habiendo sido probado por diferentes prompts, generaliza y predice muy bien el estado de ánimo en varios contextos. En un principio se tuvo un problema con la consigna, el cual nos llevó a utilizar un modelo pre entrenado de análisis de sentimientos, aún así este modelo era poco certero en español y para este tópico en comparación con el modelo entrenado posteriormente.

Por otro lado, la recomendación de películas/libros/juegos de mesa, aun teniendo datasets acotados y la falta de alguna que otra tecnología, en general es acertada, aunque tenga sus ejemplos donde le es complicado encontrar recomendaciones acorde a la preferencia, donde se piensa que puede ser problema de una preferencia muy simple, teniendo similitudes con demasiados embeddings, la falta de material el cual pueda ser recomendado o la afinidad de la descripción/género con la película/libro/juego de mesa. En esta parte del trabajo se dificultó la integración del NER a la búsqueda semántica, primero siendo que no identificaba por completo nuestro requisito de entidades (géneros, sentimientos y algunos autores), también existía el caso en que la persona quisiera obviar ciertos temas, negando la preferencia y unas de las etiquetas a identificar eran la palabra “película” , “libro” y “juego de mesa”. Concluyendo que se utilizará para identificar personas como actores, autores y directores. Un inconveniente que no tiene relevancia en cómo funciona la recomendación o que devuelven los modelos es que la recomendación y la clasificación se muestran 2 veces, pensamos que es una vez por el print normal de python y otra por la librería IpyWidgets que no funciona de otra manera.

Aún así, el algoritmo cumple con la consigna, siendo que puede clasificar el ánimo de una persona en base a un prompt ingresado por el mismo, puede dar un recomendación en base a un prompt de un usuario, y además se le agregó una interfaz sencilla para poder tener más opciones de búsqueda por parte del usuario.

Se recomienda para trabajos futuros delimitar un poco la libertad del trabajo, ya que aunque esté bueno esta idea de que en un escenario real podría ser de esta manera, tener una consigna un poco más marcada podría ayudar a que se tengan menos dudas y malas lecturas de la consigna. En nuestro caso no se llegó a entender la idea de que el clasificador tenía que ser entrenado por nosotros en un principio y también no se entendía la relación entre las consignas.

ANEXOS

- https://arxiv.org/pdf/1908.10084 (Paper del modelo utilizado para los embeddings).
- https://github.com/nacho-gonz/NLP-Entregas-2024 (Repositorio para la entrega de la materia)