Modelo de Lenguaje y Sistema de Recomendación

Se desea crear un sistema para recomendar serie de anime. El archivo `anime.csv` poseen una base de datos donde usuarios calificaron diferentes animes (-1 significa sin calificar).

(**a**) *Modelo de Lenguaje:* Se desea diseñar un buscador de títulos de películas, de manera que si el usuario comete algún error u omisión cuando lo escribe, el buscador pueda entender.

1. Cargar la base de datos `anime.csv`.
2. Cargar el modelo de lenguaje. : Posible código (adaptar a sus necesidades):

   ```
   from sentence_transformers import SentenceTransformer
   model = SentenceTransformer('sentence-transformers/paraphrase-multilingual-mpnet-base-v2')
   ```

   El modelo de lenguaje posee una arquitectura *transformer* llamada XLM-Roberta. En lugar de trabajar palabra por palabra, este transformer utiliza *subword tokens* (utilizando la raíz de las palabras con una perspectiva semántica).
3. Incorporar al modelo de lenguaje un procesamiento de texto que convierta mayúsculas en minúsculas y unifique espacios en blanco.
4. Computar el embedding de "Attack on Titan". Indicar su dimensión.
5. Se desea medir que tan parecidos son dos *embeddings*. Para ello, implementar un código que calcule la *similitud coseno*. : La similitud coseno se define como el coseno del ángulo entre dos vectores $\mathbf{SC}(u, v) = \frac{u \cdot v}{\|u\|\|v\|}$.
6. Implementar un buscador que, dado un *string* (a partir de su correspondiente *embedding*), devuelva las k series con una representación más similar.
7. Para verificar la característica semántica de los *subword tokens*, buscar la palabra "Giant" con $k = 15$. Justificar resultados desde la perspectiva semántica.
8. Encapsular todo el buscador en una clase.

(**b**) *Sistema de Recomendación:* Se desea diseñar el sistema de recomendación y utilizarlo para recomendarnos películas.

1. Hallar, analíticamente, el gradiente del *riesgo empírico regularizado* correspondiente a un filtro colaborativo. Expresar los resultados vectorizados.
2. Agregar un usuario a la base de datos con al menos 10 series calificadas. Utilice el buscador para no tener que escribir los títulos perfectos.
3. Utilizando *gradiente descendente* entrenar un filtro colaborativo con un espacio latente de dimensión 15, $\lambda = 10$ y *learning rate* 10^{-3}. Graficar el riesgo regularizado empírico en función del número de iteraciones (al menos 2000). : Una normalización razonable puede ser llevar los valores numéricos al $[0, 1]$.
4. Crear un *rating* ponderando en partes iguales la salida del filtro colaborativo y la calificación media de las películas.
5. Recomendar las 10 películas **no vistas** con más alto *rating* al usuario creando anteriormente.