PRACTICA 2: Análisis de información textual en YouTube (4 puntos)

En los últimos años, YouTube se ha convertido en un pilar fundamental para la generación de contenidos y la realización de estudios basados en el análisis del lenguaje natural, tanto en el ámbito social como en el académico. Ejemplos de dichos estudios incluyen el análisis de la reputación de canales, la evaluación del impacto de estrategias de contenido, la extracción de opiniones y la predicción de tendencias dentro de la plataforma.

Los factores que contribuyen a la popularidad de esta plataforma en el contexto del Procesamiento del Lenguaje Natural (PLN) son los siguientes: 1) es una plataforma abierta con una API que facilita el acceso a los contenidos; 2) la plataforma permite dar una descripción en texto de los vídeos; 3) cada vídeo suele tener un conjunto de comentarios sobre el mismo que puede servir para hacer un análisis de sentimientos efectivo.

En esta práctica extraeremos contenido de YouTube, procesaremos la información textual de los vídeos extraídos y realizaremos experimentos de clasificación y análisis de sentimientos.

Pasos a seguir:

1) Extracción de datos de YouTube (calificación 0,75)

1.1) En primer lugar, utilizaremos la API de YouTube a través de la librería google-api-python-client en Python. Por tanto, el primer paso consiste en instalar esta librería y familiarizarse con su documentación (https://developers.google.com/youtube/v3).

1.2) Para poder acceder a la API de YouTube, es necesario obtener una clave de API de Google. Si no tienes una, puedes obtenerla siguiendo las instrucciones en la Consola de Desarrolladores de Google (https://console.developers.google.com/). Una vez obtengas la clave de API, esta se usará en el código para autenticar las solicitudes a la API de YouTube y acceder a los contenidos del canal.

Es importante tener en cuenta las políticas de uso de la API de YouTube. Asegúrate de revisar los términos y condiciones establecidos por YouTube para desarrolladores. Esto incluye limitaciones en el número de solicitudes que se pueden hacer a la API en un cierto período de tiempo. Asegúrate de respetar estos límites para evitar restricciones en el acceso a la API. Además, debes seguir las prácticas recomendadas para el manejo de datos y el respeto de la privacidad de los usuarios en la plataforma.

1.3) El primer paso del proyecto consiste en recopilar una cantidad significativa de contenido de YouTube para su posterior análisis. Para ello, se recomienda elegir un tema a vuestra elección y descargar tanto contenido de varios tipos de canales de ese tema. Por ejemplo, podemos descargar videos sobre bricolaje, películas, viajes, fútbol, etc. De cada tipo de canal (bricolaje, películas, etc), habría que seleccionar un mínimo de 10 canales y, de cada canal concreto un mínimo de 100 vídeos. Para cada vídeo se deberá descargar la información sobre el título, la descripción del vídeo y un conjunto de comentarios del vídeo que se está procesando. Además, se deberían seleccionar al menos 3 tipos de canales con lo que el número total de información sobre vídeos debe ser de al menos 3*10*100. Es muy recomendable, además, sacar información de vídeos variados que estén distribuidos en el tiempo; es decir, que no sean todos del mismo día y que, en la descripción, se hable de tópicos distintos, pero relacionados con el canal. El criterio para seleccionar los vídeos (a mano, con alguna

heurística, etc.) queda a decisión del alumno. Sin embargo, esta decisión influirá mucho en la calidad de los modelos generados al final.

Para almacenar la información recopilada, se recomienda crear archivos JSON individuales para cada canal o personalidad.

ejemplo_canal.json

```
{
  "channel": "NombreDelCanal",
  "type": "tecnología",
  "videos": [
    {
      "date": "2023-10-25 15:30",
      "title": "Últimas novedades en gadgets tecnológicos",
      "description": "En este video, exploramos las últimas innovaciones en...",
      "comments": [
        {
          "user": "Usuario1",
          "comment": "Interesante información sobre...",
          "sentiment": ""
        },
        {
          "user": "Usuario2",
          "comment": "No estoy de acuerdo con la afirmación sobre...",
          "sentiment": ""
        }
      ]
    }
  ]
}
```

Sobre estos formatos, iremos más adelante añadiendo más datos, como el sentimiento de cada comentario en cada vídeo.

2) Clasificador del tipo de canal (calificación 0,75)

La segunda tarea consiste en crear un clasificador del tipo de canal. A modo general, este modelo será capaz de predecir el tipo de canal a partir de una descripción del vídeo individual. Por ejemplo, pasándole una descripción de un vídeo, el modelo te dirá si pertenece a un canal de bricolaje, cocina o fútbol.

En este tipo de experimentos, un paso crucial es realizar una correcta separación de los datos que se usarán para entrenar el modelo y cuáles se usarán para validar el modelo. En este caso concreto, una estrategia sería nunca usar descripciones de vídeos de un canal que aparece en el entrenamiento para validar, porque el modelo podría aprender erróneamente a clasificar a un canal porque ha visto comentarios suyos al entrenar. Por ejemplo, si tuviéramos vídeos de un canal determinado de bricolaje tanto en el entrenamiento como en la validación del modelo, el modelo estaría dando por bueno todos aquellos patrones que formen parte de la escritura de ese canal, como coletillas que use para escribir, pero que sean cosas que no compartan otros canales de bricolaje y, por lo tanto, el modelo no está generalizando, sino aprendiendo un sesgo.

Una separación de entrenamiento y validación podría ser: 70% y 30%. Por ejemplo, si tenemos 3 tipos de canales, y 10 canales por cada tipo de canal, tendríamos

- 3 * 7 * 100 descripciones de vídeos para entrenar
- 3 * 3 * 100 descripciones de vídeos para validar

Otro aspecto importante es que el canal se analiza a nivel de cada descripción del vídeo, y se clasifica de manera global. Esto quiere decir que una vez tenemos el modelo. Si queremos saber qué tal se comporta, seleccionamos un canal que no se haya utilizado en el entrenamiento, y miramos cuantos de las descripciones de los vídeos se clasifican como "bricolaje". Si de 100 vídeos acierta la mitad, el acierto del modelo sería de un 50%.

Para representar las descripciones de los vídeos hay que seleccionar algún tipo de representación. Como una primera aproximación, vamos a usar modelos estadísticos, tales como la bolsa de palabras BoW. Para ello, vamos a usar herramientas de Scikit-learn tales como CountVectorizer y TfidfVectorizer. Estas herramientas nos permitirán representar cada texto como un conjunto de vectores. Es importantísimo aprender estos vectores sólo del conjunto de entrenamiento y dejar los de test aparte.

Para construir el modelo, podéis emplear algún modelo de clasificación, tal como Random Forest o Support Vector Machines (SVM). La elección del modelo es libre. También es necesario para sacar la máxima nota en este apartado probar varios modelos de clasificación para ver el desempeño de estos modelos.

Se debe hacer un pequeño análisis de los resultados. Para ello, una vez obtenidos los resultados se pueden sacar las matrices de confusión[1] y un informe de resultados[2].

3) Extracción del sentimiento de los comentarios de los vídeos (calificación 0,75)
En esta tarea, vamos a emplear modelos del lenguaje ya entrenados para poder extraer la polaridad subjetiva de cada tweet. Para ello emplearemos un modelo basado en BERT (BETO) que utiliza word embeddings contextuales. Los word embeddings son representaciones densas que palabras que codifican palabras como vectores, de manera que se pueden agrupar palabras con significados parecidos. Además, los word embeddings contextuales permiten que una palabra tenga una codificación distinta según su contexto; es decir, las palabras que lo acompañan. De esta manera, el texto "He ido al banco a sacar dinero y me he sentado en el banco de enfrente", las palabras "banco" tienen representaciones diferentes.

Vamos a usar un modelo pre-entrenado de BERT para análisis de sentimientos en español, almacenado en el repositorio de HuggingFace:
https://huggingface.co/finiteautomata/beto-sentiment-analysis

Este modelo ha sido ajustado para predecir sentimientos positivos, negativos y neutrales a partir de un dataset en español llamado TASS 2020 (Vega et al., 2020).

Un ejemplo de uso de este modelo se vio en la *sesión 6 de prácticas*.

El sentimiento de los comentarios se almacenará de nuevo en los ficheros .JSON en el campo "sentiment" de cada comentario. Es importante entregar esos ficheros JSON modificados para valorar correctamente este apartado.

4) Búsqueda de canales similares (calificación 0,75)
Aplicar técnicas de similitud de documentos usando word embeddings de fastText para encontrar cuáles son los perfiles más parecidos. La idea aquí es poder encontrar cuáles son los perfiles más parecidos entre sí, de acuerdo con la descripción de sus vídeos, y poder emplear

[1] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html
[2] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html

herramientas de visualización para poder distinguir cuáles son los canales que encajan y que no encajan con su grupo. Para ello, se concatenarán todos las descripciones de los vídeos del mismo canal y se obtendrá un sentence embeddings. Se recomienda el uso de la librería fastText para obtener estos vectores tal y como se vio en la *sesión 5 de clase de prácticas*. También se recomienda usar los vectores de 300 dimensiones de fastText en español[3].

5) Tareas optativas (total, 1 punto)
5.1) Analizar distintos modelos y realizar un análisis comparativo de distintos tipos de características (unigramas, bigramas, combinaciones, char n-gramas, etc). (calificación 0,3 puntos). Se replicará el apartado 4 (clasificación del tipo de canal) utilizando distintas características de unigramas, bigramas, combinaciones, char n-gramas etc y se discutirá un poco los resultados.

5.2) Aplicar técnicas de pre-procesamiento para la limpieza de los datos para la clasificación del tipo de perfil (calificación 0,3 puntos). Para ello, se pueden eliminar URLs, menciones, y otras partes del texto que parezcan no relevantes y ver si se mejora o no la clasificación. También se pueden sustituir estos elementos por tokens fijos. Otra idea es la corrección ortográfica de palabras, o el aplicar procesos de stemming para simplificar el vocabulario.

5.3) Aplicar el uso de Transformers para la clasificación del tipo de perfil (calificación 0,4 puntos). Se hará un finetuning de un modelo destilado de RoBERTa en español (https://huggingface.co/dccuchile/albert-base-spanish) para la clasificación del tipo de canal similar al que se hizo en la *sesión 6 de clase de prácticas*.

Entregables:
1) Guiones python (.py)
2) Python Notebooks (.pynb) (sed particularmente cautos/as en detallar los resultados de los experimentos, etc)
3) Enlace para la descarga de los documentos generados (.zip)

Es fundamental que los Notebooks sea auto explicativo de todos los pasos (con celdas textuales acompañando a celdas con código y que contenga explícitamente los resultados -sin tener que ejecutar las celdas de nuevo-). Comprobad esto antes de enviar el Notebook. Cualquier proyecto de Analítica de Datos debe ser auto documentado y sus experimentos fáciles de reproducir. Un aspecto clave en la evaluación de esta práctica reside en la calidad de las explicaciones y documentación que acompañéis al código dentro del Notebook.

- Valoración y Fecha de Entrega:
  - Esta práctica tiene una valoración de 4 puntos (3+1 opcional) (sobre el total de 7 puntos de la parte práctica de la materia)

Fecha entrega: 20 de enero a las 23:55h

Se recomienda entregar la práctica en cuanto la tengáis terminada para que pueda corregirse y evaluarse por el profesor cuanto antes.

[3] https://dl.fbaipublicfiles.com/fasttext/vectors-crawl/cc.es.300.bin.gz

Referencias

Vega, M. G., Díaz-Galiano, M. C., Cumbreras, M. Á. G., Arco, F. M. P. del, Montejo-Ráez, A., Zafra, S. M. J., Cámara, E. M., Aguilar, C. A., Cabezudo, M. A. S., Chiruzzo, L., & Moctezuma, D. (2020). Overview of TASS 2020: Introducing Emotion Detection. En M. Á. G. Cumbreras, J. Gonzalo, E. M. Cámara, R. Martínez-Unanue, P. Rosso, S. M. J. Zafra, J. A. O. Zambrano, A. Miranda, J. P. Zamorano, Y. Gutiérrez, A. Rosá, M. Montes-y-Gómez, & M. G. Vega (Eds.), *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020) co-located with 36th Conference of the Spanish Society for Natural Language Processing (SEPLN 2020), Málaga, Spain, September 23th, 2020* (Vol. 2664, pp. 163-170). CEUR-WS.org. http://ceur-ws.org/Vol-2664/tass_overview.pdf