Desarrollo de métodos de visualización para análisis de información multimodal

Informe Técnico de Beca de Estancia de Investigación en el verano

Presenta:

Rodrigo Torrealba Cruz

Asesor:

Dr. Hugo Jair Escalante Balderas

Resumen

Este proyecto se enfoca en la extracción y análisis de características de videos. Iniciando con la preparación del entorno de desarrollo y descargando recursos esenciales. Luego, se extraen características clave de audio y video de los videos mediante modelos pre-entrenados.

La parte central del proyecto se concentra en analizar grupos de videos con base en estas características utilizando algoritmos de agrupación como K-Means. Se evalúa la calidad de los grupos mediante métricas específicas y se realiza una visualización bidimensional de la distribución de los videos. Además, se explora una variante que examina las características de los videos en RGB, Flow y Speech por separado, aplicando técnicas similares de agrupación y reducción de dimensionalidad.

En última instancia, este proyecto proporciona una perspectiva general sobre cómo abordar la extracción y análisis de características de videos, así como la agrupación y evaluación de estos videos en clústeres. Estos conocimientos pueden ser valiosos en aplicaciones como recomendación de contenido y clasificación automática de videos.

Palabras Clave: Procesamiento de Video, Extracción de Características, K-Means, reducción de dimensionalidad, Agrupación No Supervisada.

Introducción

El problema del discurso de odio en las redes sociales es una preocupación creciente en nuestra sociedad contemporánea. A medida que las plataformas digitales se han vuelto omnipresentes en nuestras vidas, el intercambio de ideas y opiniones se ha acelerado y amplificado enormemente. El anonimato relativo y la distancia física que brindan las redes sociales permiten a las personas expresar libremente sus prejuicios y promover la intolerancia, el racismo, la xenofobia, la homofobia y otros tipos de discriminación. Este tipo de discurso no solo es dañino para los individuos y comunidades directamente afectados, sino que también socava los fundamentos mismos de una sociedad democrática y pluralista. Por lo que es imperativo el desarrollo de herramientas tecnológicas que permitan identificar y erradicar este tipo de discurso.

Lo anterior aunado a la implementación de políticas y mecanismos eficaces para identificar, denunciar y prevenir el discurso de odio en las redes sociales. Por otro lado, las redes sociales se han diversificado enormemente en los últimos años, dando lugar a todo tipo de modalidades de información en las que el discurso de odio puede manifestarse. Por tanto, herramientas tecnológicas para abordar esta problemática deben tener la capacidad de procesamiento de información multimodal.

Objetivo principal

El objetivo principal de este proyecto se centra en el desarrollo de métodos de visualización para el análisis de información multimodal. Esta tarea involucra la exploración y comprensión de datos que provienen de diversas fuentes y modalidades, como el análisis de video y audio en conjunto. La visualización desempeña un papel fundamental en la interpretación de datos complejos y en la identificación de patrones y tendencia.

Actividades

- **Familiarización y Formación:** Realizar reuniones con estudiantes y profesionales para discutir la problemática y objetivos del proyecto. Tomar tutoriales y recursos para familiarizarse con los métodos de aprendizaje de representaciones de información multimodal.
- **Representación Multimodal:** Investigar y seleccionar métodos adecuados para la representación de los videos en las colecciones consideradas. Implementar los métodos elegidos y realizar pruebas preliminares para evaluar su efectividad.
- **Reducción de Dimensionalidad y Visualización:** Desarrollar e implementar métodos de reducción de dimensionalidad para los datos multimodales obtenidos. Explorar técnicas de visualización para representar los datos reducidos en formatos interpretables.
- **Visualización Avanzada y Análisis:** Refinar las técnicas de visualización para resaltar patrones relevantes en los datos. Analizar visualizaciones generadas y extraer conclusiones preliminares sobre las características de los videos asociados al contenido cuestionable y discurso de odio.
- **Agrupamiento y Análisis Adicional:** Aplicar métodos de agrupamiento a los datos para identificar clústeres o patrones de comportamiento. Realizar un análisis más profundo de los grupos identificados y documentar hallazgos relevantes..
- **Refinamiento y Pruebas:** Realizar refinamientos en los métodos implementados según la retroalimentación recibida. Realizar pruebas para validar la efectividad y robustez de los métodos de análisis desarrollados.
- **Documentación y Preparación del Informe:** Generar un repositorio completo que incluya el código, los recursos y las visualizaciones generadas. Preparar el informe final que documente de manera detallada los métodos, los resultados y las conclusiones del proyecto.

Resultados esperados

Derivado de la estancia del estudiante se espera obtener un documento donde se reporte el análisis realizado con las técnicas implementadas. De igual forma se espera que al terminó de la estancia se cuente con código que permita replicar los experimentos realizados por el estudiante. Toda esta información estará disponible en un repositorio.

Metodología

Adquisición de Datos

El conjunto de datos utilizado en este proyecto, conocido como "ComicMischief Dataset" Este conjunto de datos contiene anotaciones de contenido de clasificación multiclase para cada escena de video. Estas anotaciones se aplican a nivel de escena y no se asocian a una modalidad específica. Además de las anotaciones, el conjunto de datos incluye meta-datos importantes sobre los videos, como detalles sobre su origen y contenido.

Este conjunto de datos se compone de videos recopilados de fuentes variadas, incluyendo sitios web como YouTube y IMDB, y abarca una amplia gama de contenido relacionado con la comedia. Con el fin de analizar y comprender mejor estos videos, se han identificado y etiquetado cuatro categorías de contenido específicas: Sarcasmo, Humor Físico, Humor Sangriento y Humor para Adultos. Estas categorías nos permiten clasificar y estudiar los patrones de comedia presentes en los videos con mayor detalle, lo que es esencial para los objetivos de este informe.

Extracción de Características

Extracción de Características de Audio - VGGish:

En el contexto de la metodología empleada, se emplea el modelo VGGish para realizar la extracción de características de audio de los videos. VGGish es un modelo que transforma señales de audio, tales como las provenientes de un flujo de video, en representaciones numéricas que capturan atributos acústicos significativos. Esto se logra a través de varios procesos que incluyen la conversión de señales de audio en espectrogramas y la posterior traducción de estos espectrogramas en vectores numéricos. En esencia, VGGish actúa como un extractor automático de características de audio, lo que facilita la comprensión y análisis del contenido sonoro presente en los videos.

Extracción de Características de Video - I3D:

Por otro lado, en la extracción de características visuales de los videos, se utiliza el modelo I3D (Inflated 3D Convnet), el cual ha sido pre-entrenado en conjuntos de datos visuales de gran envergadura. El propósito principal de I3D es discernir información relevante de los fotogramas de video, incluyendo la detección de objetos, movimiento y atributos visuales diversos. Por ejemplo, en el caso de un video que muestra una pelota en movimiento, el modelo I3D puede identificar la presencia de una pelota, su color y la dinámica de su movimiento en los diferentes fotogramas. Estos atributos visuales son luego convertidos en vectores numéricos que representan de manera concisa y cuantitativa dichas características.

Figura 1 - Representación vectorial del contenido

Preprocesamiento de Datos

Escalado de Características con StandardScaler:

Antes de embarcarnos en el análisis y agrupación de características, es esencial preparar los datos para garantizar la coherencia y la comparabilidad entre las diferentes características extraídas de los videos. Una de las etapas fundamentales de este proceso es el escalado de características, que se lleva a cabo mediante el empleo de la técnica estandarización. Esta técnica es crucial para estandarizar las características numéricas, asegurando que todas ellas tengan una media de cero y una desviación estándar de uno. De esta manera, se eliminan las disparidades en la escala de las características, lo que podría distorsionar los resultados del análisis de agrupación. El uso de StandardScaler crea una base uniforme sobre la cual se pueden aplicar técnicas de agrupación y evaluación, permitiendo un análisis más preciso y significativo de los datos.

Reducción de Dimensionalidad con t-SNE:

La reducción de dimensionalidad es otra fase clave en la preparación de datos y juega un papel vital en la simplificación de conjuntos de características complejas. Para lograr esto, se emplea la técnica t-Distributed Stochastic Neighbor Embedding (t-SNE). t-SNE es un algoritmo de reducción de dimensionalidad no lineal que se utiliza para representar datos de alta dimensionalidad en un espacio bidimensional o tridimensional de manera que se preserven las relaciones entre los puntos. En el contexto de este proyecto, t-SNE se aplica a las características previamente escaladas para crear representaciones visuales que simplifican la estructura de los datos. Esto puede revelar patrones subyacentes y relaciones entre los videos que de otro modo serían difíciles de percibir en el espacio de características original de alta dimensión. La utilización de t-SNE proporciona una herramienta invaluable para la visualización y la comprensión de la estructura intrínseca de los datos antes de llevar a cabo la agrupación y la evaluación de clústeres.

Clustering

En este proyecto, se emplea el algoritmo de agrupación K-Means para realizar la tarea de clustering sobre las características extraídas de los videos. K-Means es una técnica ampliamente utilizada en el análisis de datos que tiene como objetivo dividir un conjunto de datos en grupos o clústeres, donde cada punto de datos pertenece al clúster cuyo centroide (punto central) es el más cercano. La elección de K, el número de clústeres, es un paso crítico en el proceso de aplicación de K-Means y puede afectar significativamente los resultados del clustering. En este proyecto, se exploran diferentes valores de K para evaluar cómo varían los clústeres en función de esta elección. Una vez que se determina el número óptimo de clústeres, el algoritmo K-Means se ejecuta para agrupar los videos en función de sus características similares. Este enfoque permite la identificación de grupos de videos con características compartidas, lo que puede ser fundamental para aplicaciones como la recomendación de contenido o la clasificación automática de videos. Este proceso de clustering con K-Means es una etapa esencial en la comprensión y organización de los datos de video extraídos, proporcionando una base sólida para el análisis posterior.

Evaluación del Clustering:

La etapa de evaluación desempeña un papel fundamental en la metodología implementada para comprender la calidad y la coherencia de los clústeres generados mediante el algoritmo K-Means. Esta evaluación se basa en un conjunto selecto de métricas que ofrecen una perspectiva objetiva del rendimiento del proceso de agrupación. Las métricas de evaluación utilizadas en este estudio incluyen:

Silhouette Score: Este indicador cuantifica la cohesión intraclúster y la separación interclúster. Un valor más alto del Silhouette Score indica una mejor calidad de clustering.

Calinski-Harabasz Score: También conocido como el índice de variedad entre clústeres, este puntaje mide la relación entre la dispersión entre clústeres y la dispersión dentro de los clústeres. Un puntaje superior sugiere una mejor separación entre los grupos.

Davies-Bouldin Score: Esta métrica evalúa la similitud promedio entre cada clúster y su clúster más similar. Un valor inferior en el Davies-Bouldin Score señala una mejor separación entre los clústeres.

Además de estas métricas estándar, también se incorporan otras métricas adicionales, como el Índice de Rand y la Purity, que contribuyen a una evaluación más completa del rendimiento del clustering. El Índice de Rand mide la similitud entre los clústeres obtenidos y las etiquetas de clase reales (si están disponibles), mientras que la Purity refleja la proporción de instancias correctamente asignadas a sus respectivos clústeres. En conjunto, estas métricas proporcionan una evaluación exhaustiva de la calidad y la efectividad del proceso de agrupación de videos en función de sus características intrínsecas.

Visualización de resultados

La visualización de resultados se lleva a cabo utilizando una técnica de reducción de dimensionalidad llamada t-SNE (t-Distributed Stochastic Neighbor Embedding). Esta técnica permitie proyectar las características de los videos en un espacio bidimensional, lo que facilita la visualización de la distribución de los videos en función de sus similitudes y diferencias intrínsecas.

Para crear visualizaciones efectivas, se emplean herramientas y bibliotecas de Python, como Matplotlib. Estas bibliotecas posibilitan la creación de gráficos y diagramas que representan la distribución de los videos en el espacio t-SNE.

Se generan diagramas de dispersión que muestran la disposición de los videos en el espacio bidimensional. Cada punto en el diagrama de dispersión representa un video, y se utilizan colores y símbolos distintivos para resaltar clústeres específicos o patrones de agrupación. Esta visualización ayuda a identificar agrupaciones naturales y relaciones entre los videos.

Desarrollo

Preparando el Entorno

Se instala Miniconda y se configura el entorno necesario para el proyecto. Se instalan las bibliotecas requeridas, incluyendo algunas bibliotecas de procesamiento de video.

```
[ ]  !wget https://repo.anaconda.com/miniconda/Miniconda3-py37_23.1.0-1-Linux-x86_64.sh -q
     !bash ./Miniconda3-py37_23.1.0-1-Linux-x86_64.sh -b -f -p /usr/local

[ ]  import locale
     locale.getpreferredencoding = lambda: "UTF-8"
     !pip install cryptography==38.0.4
     !pip install pyopenssl --upgrade

[ ]  import os
     from pathlib import Path
     import sys
     sys.path.append('/usr/local/lib/python3.7/site-packages/')
     from sample.single_video_prediction import get_video_duration

[ ]  # Extracion de caracteristicas
     !export PIP_DEFAULT_TIMEOUT=100
     !conda env create -f ./submodules/video_features/conda_env_i3d.yml
     !conda env create -f ./submodules/video_features/conda_env_vggish.yml
     # Modelo de lenguaje Spacy
     !/usr/local/envs/bmt/bin/python -m spacy download en
```

Figura 2 - Preparación del entorno

Descargando los Checkpoints

Se descargan y se colocan en una ubicación especifica los checkpoints pre-entrenado necesarios para las tareas de procesamiento de video

```
[ ]  !wget https://a3s.fi/swift/v1/AUTH_a235c0f452d648828f745589cde1219a/bmt/glove.840B.300d.zip -q
     !wget https://storage.googleapis.com/audioset/vggish_model.ckpt -q --show-progress

     !mkdir .vector_cache
     !mv glove.840B.300d.zip ./.vector_cache/
     !mv vggish_model.ckpt ./submodules/video_features/models/vggish/checkpoints/
```

Figura 3 - Descarga de modelos pre-entrenados

Cargando la Ruta a los Videos

Se monta Google Drive en el entorno Colab y se establece la ruta donde se encuentran los videos que serán procesados.

```
[ ]  from google.colab import drive
     drive.mount('/content/drive')
```

```
[ ]  # Directorio que contiene los videos
     video_directory = '/content/drive/MyDrive/HSD/HSD_Videos/'

     # Extensión de archivo
     file_extension = '.mp4'

     # Lista para almacenar las rutas de los videos
     VIDEO_PATHS = []

     # Recorre los archivos en el directorio
     for filename in os.listdir(video_directory):
         if filename.endswith(file_extension):
             video_path = os.path.join(video_directory, filename)
             VIDEO_PATHS.append(video_path)
```

Figura 4 - Configurando ruta de videos y directorio de almacenamiento de características

Extracción de Características

Se recorren los videos en la ruta especificada y se extraen las características de video utilizando diferentes modelos (I3D y VGGish). Las características se almacenan en un directorio especificado para su posterior análisis.

```
for video_path in VIDEO_PATHS:
    # Preparar las rutas de salida para las características de audio y video
    VIDEO_DURATION = get_video_duration(video_path)
    FEATURES_PATH_STUB = os.path.join(FEATURES_CACHE_PATH, Path(video_path).stem)
    FEATURE_PATH_VGGISH = f'{FEATURES_PATH_STUB}_vggish.npy'
    FEATURE_PATH_RGB = f'{FEATURES_PATH_STUB}_rgb.npy'
    FEATURE_PATH_FLOW = f'{FEATURES_PATH_STUB}_flow.npy'


    # Extraer características de video
    !cd ./submodules/video_features && /usr/local/envs/i3d/bin/python main.py \
        --feature_type i3d \
        --on_extraction save_numpy \
        --device_ids 0 \
        --extraction_fps 25 \
        --video_paths {video_path} \
        --output_path {FEATURES_CACHE_PATH}

    # Extraer características de audio
    !cd ./submodules/video_features && /usr/local/envs/vggish/bin/python main.py \
        --feature_type vggish \
        --on_extraction save_numpy \
        --device_ids 0 \
        --video_paths {video_path} \
        --output_path {FEATURES_CACHE_PATH}
```

Figura 5 - Extracción de características

Importación de Librerías Necesarias

Se importan las bibliotecas necesarias para el análisis posterior, incluyendo NumPy, Pandas, Matplotlib y Scikit-learn.

```python
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.manifold import TSNE
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score, calinski_harabasz_score, davies_bouldin_score,
from sklearn.metrics.cluster import contingency_matrix
from IPython.display import HTML, display
```

Figura 6 - Importación de librerías

Carga de Características

Se cargan las características de video previamente extraídas en memoria. Las características se agrupan en matrices y se generan etiquetas únicas para cada característica.

```python
for video_path in VIDEO_PATHS:
    # Preparar la ruta de salida para las caracteristicas de video
    FEATURES_PATH_STUB = os.path.join(FEATURES_CACHE_PATH, os.path.splitext(os.path.basename(video_path))[0])
    FEATURE_PATH_RGB = f'{FEATURES_PATH_STUB}_rgb.npy'
    FEATURE_PATH_FLOW = f'{FEATURES_PATH_STUB}_flow.npy'
    FEATURE_PATH_SPEECH = f'{FEATURES_PATH_STUB}_vggish.npy'

    # Carga las características de video en memoria
    video_feature_rgb = np.load(FEATURE_PATH_RGB)
    video_feature_flow = np.load(FEATURE_PATH_FLOW)
    video_feature_speech = np.load(FEATURE_PATH_SPEECH)

    # Asegura que todas las matrices tengan la misma longitud en el eje 0
    min_length = min(len(video_feature_rgb), len(video_feature_flow), len(video_feature_speech))
    video_feature_rgb = video_feature_rgb[:min_length]
    video_feature_flow = video_feature_flow[:min_length]
    video_feature_speech = video_feature_speech[:min_length]

    # Concatena las características de video RGB, Flow y Speech
    video_feature = np.concatenate((video_feature_rgb, video_feature_flow, video_feature_speech), axis=1)

    # Obtener el número de características en el video
    num_features = video_feature.shape[0]

    # Extraer el número del nombre de la característica
    feature_number = int(os.path.basename(video_path).split("_")[0])

    # Crear etiquetas únicas para las características del video
    labels = [f'{feature_number}' for idx in range(num_features)]

    video_features.append(video_feature)
    video_labels.extend(labels)

# Convierte la lista de características de video en una matriz numpy
video_features = np.vstack(video_features)
```

Figura 7 - Carga y concatenación de características

Escalado y Reducción de Dimensionalidad

Las características de video se escalan utilizando StandardScaler y se reducen a dos dimensiones utilizando t-SNE. Esto permite visualizar las características en un espacio bidimensional y facilita la identificación de patrones.

```
[ ]  # Escala las características
     scaler = StandardScaler()
     video_features_scaled = scaler.fit_transform(video_features)

     # Aplica t-SNE para reducir la dimensionalidad a 2 dimensiones
     tsne = TSNE(n_components=2, random_state=0)
     video_features_tsne = tsne.fit_transform(video_features_scaled)
```

Figura 8 - Escalado y Reducción de Dimensionalidad

Análisis de Clustering

Se realiza un análisis de clustering utilizando el algoritmo K-Means. Se evalúa el rendimiento del clustering utilizando métricas como Silhouette Score, Calinski-Harabasz Score y Davies-Bouldin Score para determinar el número óptimo de clusters.

```
for i in range(minK, maxK):
    n_clusters = i
    kmeans = KMeans(n_clusters=n_clusters, random_state=0, init='random', n_init=10)
    video_clusters = kmeans.fit_predict(video_features_tsne)

    # Calcular Silhouette Score
    silhouette_avg = silhouette_score(video_features_tsne, video_clusters)
    silhouette_scores.append(silhouette_avg)

    # Calcular Calinski-Harabasz Score
    ch_score = calinski_harabasz_score(video_features_tsne, video_clusters)
    calinski_harabasz_scores.append(ch_score)

    # Calcular Davies-Bouldin Score
    db_score = davies_bouldin_score(video_features_tsne, video_clusters)
    davies_bouldin_scores.append(db_score)

    # Calcular el Rand Index Score
    rand_index = adjusted_rand_score(video_labels, video_clusters)
    rand_scores.append(rand_index)

    # Calcula la matriz de contingencia entre etiquetas reales y de clúster
    contingency_mat = contingency_matrix(video_labels, video_clusters)

    # Calcula el Purity Score
    purity = np.sum(np.max(contingency_mat, axis=0)) / np.sum(contingency_mat)
    purity_scores.append(purity)
```

Figura 9 - Evaluación de rendimiento de clustering

Visualización del Clúster

Se visualizan los clusters en un gráfico 2D, utilizando diferentes colores para representar cada clúster. Los centroides de los clusters se marcan en el gráfico.

```
# Número de clusters
n_clusters = 14

# Aplica K-Means a las características de video reducidas por t-SNE
kmeans = KMeans(n_clusters=n_clusters, random_state=0,
                init='k-means++', n_init=10)
video_clusters = kmeans.fit_predict(video_features_tsne)

# Obtiene los centroides de los clústeres
centroids = kmeans.cluster_centers_

# Visualiza los clusters en 2D con etiquetas de características
plt.figure(figsize=(15, 10))
scatter = plt.scatter(video_features_tsne[:, 0], video_features_tsne[:, 1],
                      c=video_clusters, cmap='rainbow')
plt.title('K-Means Clustering de Características conjuntas (t-SNE)')
plt.xlabel('Dimension 1')
plt.ylabel('Dimension 2')

# Agregar etiquetas de características a los puntos en el gráfico
for i, label in enumerate(video_labels):
    x, y = video_features_tsne[i, 0], video_features_tsne[i, 1]
    plt.text(x, y, label, fontsize=12)

# Marcar los centroides de los clústeres con círculos rojos
plt.scatter(centroids[:, 0], centroids[:, 1], c='red', marker='o',
            s=150, label='Centroides')

plt.legend()
plt.show()
```

Figura 10 - Visualización de características

Resultados

Los resultados obtenidos a través de este proyecto de extracción y análisis de características de videos ofrecen una visión de la naturaleza de los videos analizados. Tras la extracción de características de audio y video, se aplicaron algoritmos de agrupación para identificar patrones y estructuras dentro de la colección de videos. En esta sección, presentamos un resumen de los clústeres identificados, sus características y las métricas de calidad de agrupación correspondientes. Además, se incluyen visualizaciones que ofrecen una representación gráfica de la distribución de los videos en el espacio de características. Estos resultados son fundamentales para comprender mejor el contenido de los videos y pueden tener aplicaciones significativas en la recomendación de contenido y la organización eficiente de archivos multimedia.

Evaluación del rendimiento del clustering

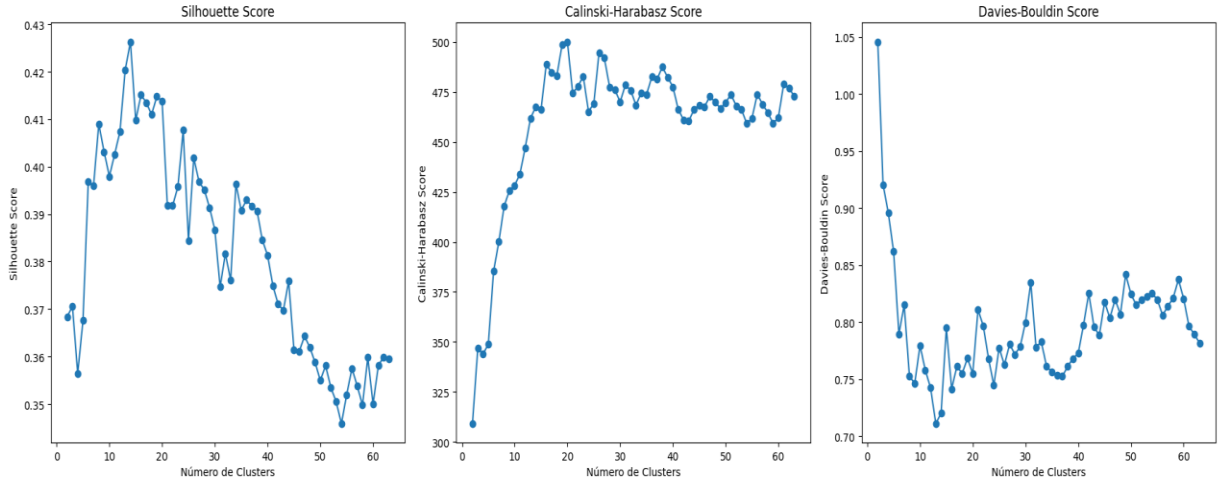

Figura 11 - Grafica de rendimiento interno de Clustering

En este análisis, hemos evaluado el rendimiento de diferentes configuraciones de clustering en un conjunto de datos utilizando tres métricas clave: Silhouette Score, Calinski-Harabasz Score y Davies-Bouldin Score. Estas métricas nos ayudan a determinar la calidad de la agrupación de datos en función del número de clústeres.

El **Silhouette Score** mide cuán bien se separan los puntos de datos en los clústeres y varía de -1 a 1. Un valor cercano a 1 indica una buena separación, mientras que valores cercanos a 0 o negativos indican que los puntos están cerca de los límites de los clusters o se han asignado incorrectamente. En nuestras gráficas, observamos que el Silhouette Score alcanza su punto más alto en el rango de 14 a 16 clusters. Esto sugiere que, según esta métrica, la estructura de los datos es más clara y coherente con estos números de clusters, lo que indica una separación efectiva.

El **Calinski-Harabasz Score** evalúa la dispersión entre los clusters en comparación con la dispersión dentro de los clusters. Valores más altos indican una mejor separación entre clusters. Nuestras gráficas revelan que el Calinski-Harabasz Score alcanza su valor máximo también en el rango de 14 a 16 clusters. Esto implica que, desde esta perspectiva, estos números de clusters proporcionan una separación efectiva y una agrupación más coherente de los datos.

El **Davies-Bouldin Score** mide la separación entre los clusters y se interpreta de manera inversa, es decir, valores más bajos indican una mejor separación. Nuestras gráficas demuestran que el Davies-Bouldin Score es mínimo en el rango de 14 a 16 clusters. Esto sugiere que, según esta métrica, estos números de clusters resultan en una separación óptima y en clusters mejor definidos.

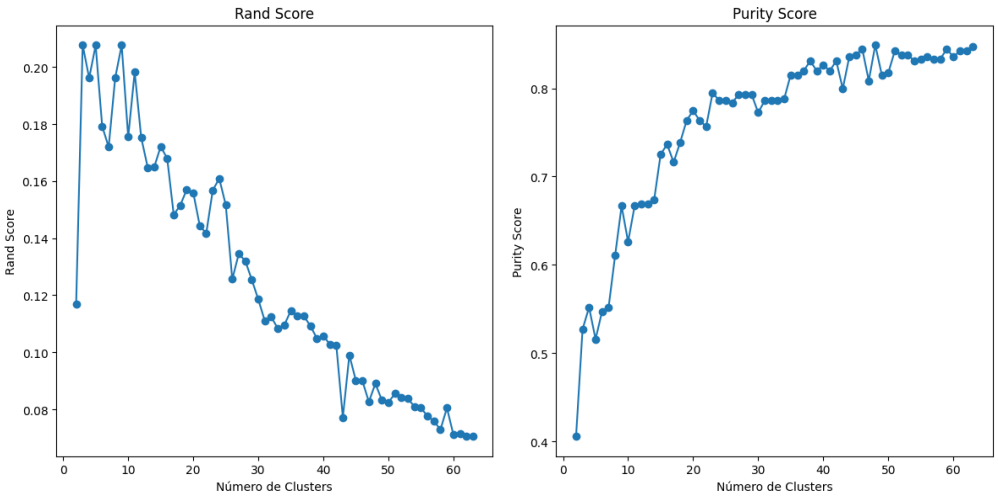

Figura 12 - Grafica de rendimiento externo de Clustering

En cuanto a las métricas externas los resultados no fueron concluyentes del todo. Debido a la cantidad de videos almacenados y su similitud entre ellos, en cuanto a clase se refiere. Los resultados del rendimiento del clustering tomando en cuenta el pre-etiquetados de los videos, nos indica que no hay una coherencia significativa entre los clústeres que se forman. Observando en las gráficas de visualización del clúster podremos obtener más información de lo que se muestra en la figura 12.

Número de Clusters	Silhouette Score	Calinski-Harabasz Score	Davies-Bouldin Score
8	0.40898430347442627	417.9524430847493	0.7526512397414076
9	0.40313857793807983	425.6537393704246	0.7461453677354064
10	0.39795100688934326	428.25420774978244	0.7794936170894984
11	0.4026036262512207	434.04857962243625	0.7579808268653107
12	0.407327800989151	447.14949421850343	0.7423735439648341
13	0.4203346073627472	461.87787517104	0.7310643634634643
14	**0.4263**	487.56168027964327	0.7200795102099392
15	0.409738689661026	4866.27368735975006	**0.714837015**
16	0.4151235818862915	**498.9260182602577**	0.7414252007669266
17	0.41350582242012024	484.64615163186636	0.7612723466206965
18	0.4109683930873871	483.24006554280174	0.7548760722855775
19	0.41474440693855286	482.8670598231312	0.7682715340576586

Tabla 1 - Rendimiento interno del Clustering

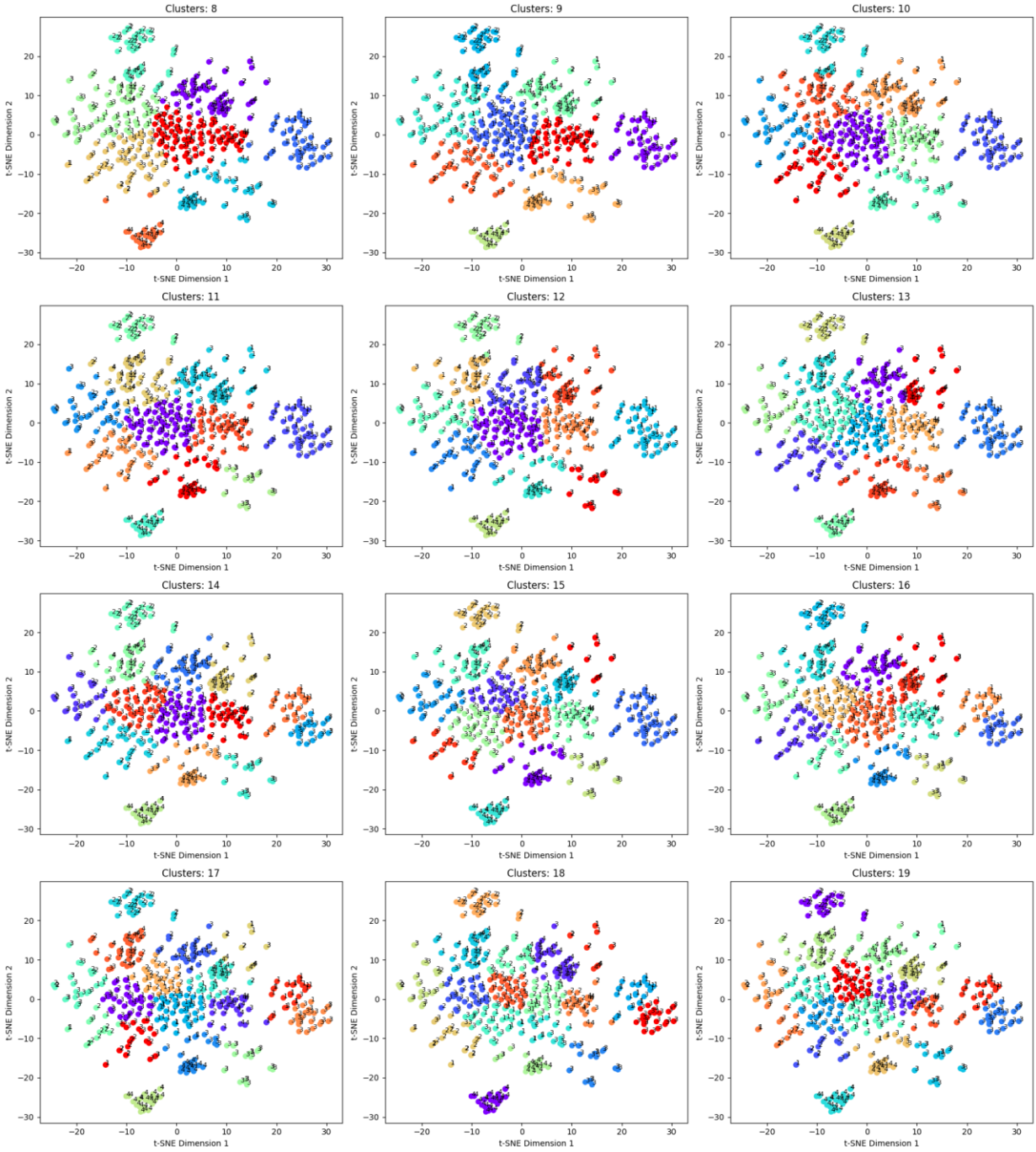

Figura 13 - Dinámica de agrupación de Clustering

Esta serie de gráficas de dispersión proporciona una instantánea de la dinámica de agrupación a lo largo de este rango, desde 8 hasta 19 clusters. Al examinar esta figura, se aprecia cómo los puntos de datos se distribuyen en función de las características seleccionadas en diferentes configuraciones de clusters. En las primeras gráficas de dispersión con un número bajo de clusters, los puntos de datos pueden parecer dispersos sin una estructura aparente. A medida que aumentamos el número de clusters, se observa cómo los puntos de datos tienden a agruparse en formaciones más coherentes y compactas.

El punto culminante en esta progresión se encuentra alrededor de los 15 clusters, donde los clusters están claramente definidos y separados, lo que indica que esta configuración es la más apropiada para la estructura subyacente de nuestros datos. Sin embargo, a medida que continuamos aumentando el número de clusters más allá de este punto, los clusters se vuelven más pequeños y fragmentados, lo que sugiere una división excesiva de los datos.

Esta figura es valiosa en el proceso de selección del número óptimo de clusters para nuestra tarea de agrupación, ya que muestra claramente cuándo los datos se organizan de manera más coherente y cuándo se vuelven demasiado fragmentados. Estas observaciones visuales complementan las métricas internas utilizadas en nuestra investigación.

La figura que muestra la progresión de clusters es una herramienta esencial que encapsula la esencia de nuestra investigación en clustering. Proporciona información crucial sobre cómo elegir el número adecuado de clusters y refleja de manera visual la complejidad y estructura de nuestros datos en diferentes configuraciones. Esta figura seguirá siendo un recurso fundamental en nuestra búsqueda continua de comprender la naturaleza de los conjuntos de datos y extraer conocimiento significativo de ellos.

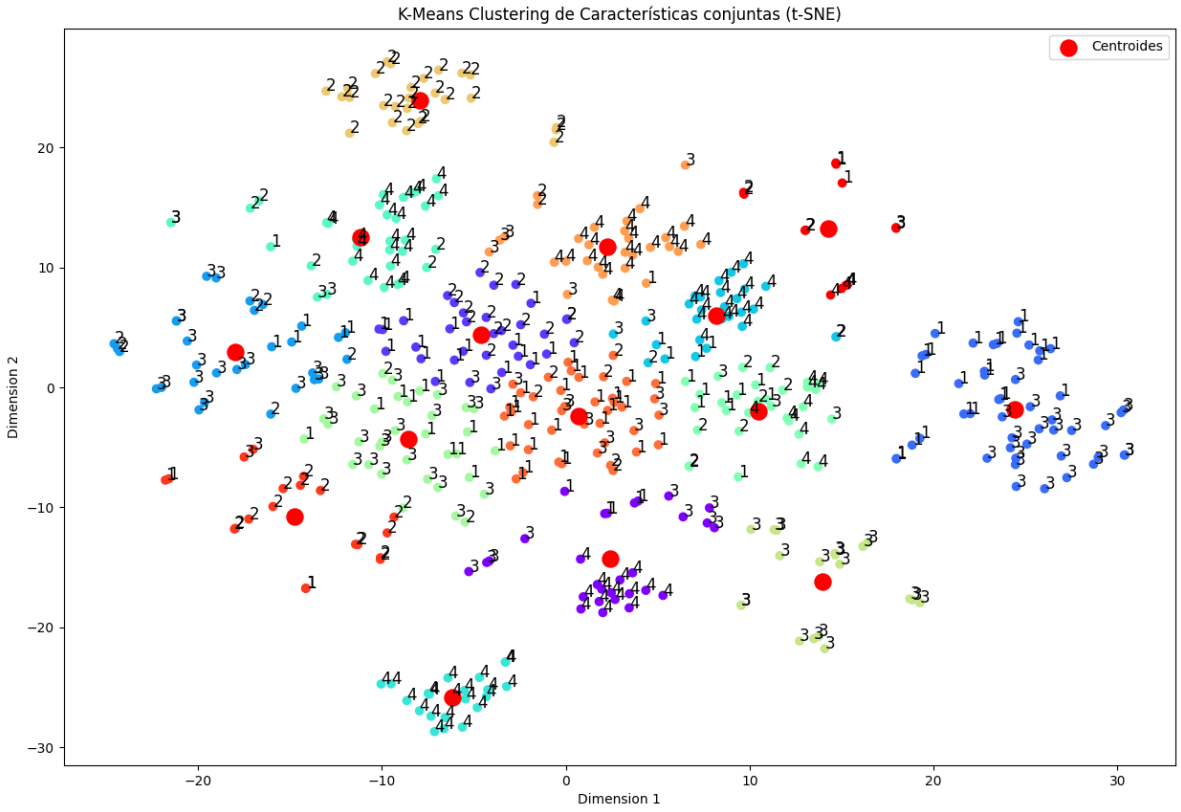

Figura 14 - Representación bidimensional de características agrupadas

Conclusiones

En el transcurso de este proyecto de clustering, hemos explorado en profundidad la tarea desafiante de agrupar datos en clusters coherentes. Nuestro objetivo principal era descubrir patrones subyacentes y relaciones significativas dentro de un conjunto de datos. A lo largo de este proceso, hemos obtenido valiosas percepciones y hemos alcanzado una serie de conclusiones esenciales.

En primer lugar, es importante destacar que, si bien nuestros resultados no son concluyentes al cien por cien en términos de la relación entre los clusters formados y las etiquetas previamente asignadas, hemos realizado avances significativos en la comprensión de la estructura de nuestros datos. En particular, al evaluar las métricas externas, hemos identificado desafíos en la coincidencia perfecta entre los clusters y las categorías predefinidas. Esto sugiere que la agrupación automática de datos puede ser una tarea compleja y que los datos pueden ser inherentemente ruidosos o ambiguos.

Sin embargo, nuestras métricas internas nos han proporcionado una visión más optimista. Hemos descubierto que un número de clusters cercano a 15 parece ser un punto óptimo para la estructura latente de nuestros datos, según indicadores como el índice de silueta, el índice Davies-Bouldin y otros criterios internos. Esto sugiere que, aunque las etiquetas originales pueden no coincidir perfectamente con los clusters encontrados, existe una estructura interna subyacente en nuestros datos que justifica la creación de estos subconjuntos.

Un aspecto destacado de nuestro proyecto ha sido la capacidad de visualizar los clusters resultantes de una manera que arroja luz sobre la organización de nuestros datos. Aunque las etiquetas originales pueden no estar distribuidas de manera óptima en los clusters, hemos identificado patrones visuales que sugieren una agrupación significativa. Esto nos lleva a la conclusión de que, a pesar de las discrepancias en las métricas externas, nuestros clusters son inherentemente informativos y reveladores.

En adelante, nuestro proyecto continuará evolucionando. Planeamos realizar modificaciones en nuestros enfoques de clustering y considerar factores adicionales que podrían estar influyendo en el rendimiento del agrupamiento. La exploración de técnicas de preprocesamiento de datos más avanzadas y la adaptación de algoritmos de clustering específicos a la naturaleza de nuestros datos son pasos lógicos para mejorar la calidad de nuestros clusters.

En última instancia, este proyecto no solo ha contribuido a nuestra comprensión de la agrupación de datos, sino que también ha enriquecido nuestro conocimiento en el campo de la analítica de datos y el aprendizaje automático. Hemos adquirido una apreciación más profunda de la complejidad inherente a la extracción de conocimiento a partir de conjuntos de datos diversos y ruidosos, y estamos emocionados por las oportunidades que se presentarán en futuras investigaciones.

Aunque nuestros resultados pueden no ser definitivos en este momento, nuestro proyecto ha sentado las bases para futuros avances y nos ha proporcionado una visión valiosa de la estructura subyacente en nuestros datos. A medida que continuamos refinando nuestro enfoque, estamos seguros de que alcanzaremos una comprensión más completa y precisa de los patrones y relaciones en nuestros datos.

Referencias

V-Iashin. (s. f.). GitHub - V-Iashin/BMT: Source code for «Bi-modal Transformer for Dense Video Captioning» (BMVC 2020). GitHub. https://github.com/v-iashin/BMT

Hassony. (s. f.). GitHub - Hassony2/kinetics_i3d_pytorch at 51240948f9ae92808c390e7217041d6fd89414e9. GitHub. https://github.com/hassony2/kinetics_i3d_pytorch/tree/51240948f9ae92808c390e7217041d6fd89414e9

Hassony. (s. f.). GitHub - Hassony2/kinetics_i3d_pytorch at 51240948f9ae92808c390e7217041d6fd89414e9. GitHub. https://github.com/hassony2/kinetics_i3d_pytorch/tree/51240948f9ae92808c390e7217041d6fd89414e9

Hugojair. (s. f.). GitHub - hugojair/mocca-at-icmi21. GitHub. https://github.com/hugojair/mocca-at-icmi21

Douglass, M. (2020). Book review: Hands-on Machine Learning with SciKit-Learn, KERAS, and Tensorflow, 2nd edition by Aurélien Géron. Physical and Engineering Sciences in Medicine, 43(3), 1135-1136. https://doi.org/10.1007/s13246-020-00913-z