CLUSTERIZACION CON TMAP

Act. 03 Análisis de Clustering con TMAP en base de datos

Análisis de Algoritmos Sección: D06
Integrantes:
López Galván Melanie Montserrat
De la Mora Villaseñor Diego Gabriel
Lopez Esparza Angel Emanuel
Maestro:
Jorge Ernesto Lopez Arce Delgado

Tabla de contenido

Introducción

La división de datos en secciones o clusters puede facilitar la interpretación o su uso en otro tipo de algoritmos. Si tenemos una población menor es más fácil enfocarse en lo que te interesa, además de requerir menos recursos a al momento de trabajar con ellos. Para observar las características de los datos y como se comportan entre ellos, se emplea la librería de TMAP (Tree-based Manifold Approximation and Projection), la cual es una técnica de reducción de dimensionalidad cuyo propósito principal es representar datos complejos y con numerosas características (por ejemplo, los píxeles de una imagen) en un espacio bidimensional o tridimensional de manera coherente y comprensible.

Objetivo

En esta actividad se busca realizar la división de cluster de una base de datos de imágenes sobre prendas de ropa, para realizar subclusters y mostrar las divisiones entre las distintas prendas, además de mostrarlo en un mapa. Además, se busca generar un entorno de anaconda con las librerías requeridas para correr el proyecto.

Desarrollo

Para dividir los datos en subclusters fue necesario modificar el archivo .csv de los datos originales, añadiendo una columna extra para definir el nuevo cluster de los que se dividirían. Este nuevo archivo .csv se emplearía para realizar la gráfica, usando tmap para realizar los grafos y posteriormente graficarlo en una pagina web usando la librería de faerun.

Código de división de categorías

El siguiente apartado describe cómo funcionan dos códigos de Python, "subcategorias_auto_fashion_mnist.py" y "unirSubcategorias.py". El objetivo de estos es unificar,

organizar y clasificar información que viene de varios archivos CSV que representan subcategorías del conjunto de datos Fashion MNIST.

Los dos scripts fueron creados con la biblioteca Pandas, además de los módulos estándar glob y os, para llevar a cabo la lectura, el manejo y la consolidación de datos.

Ambos programas comienzan leyendo todos los archivos .csv presentes en la carpeta especificada para cada archivo se realiza una lectura utilizando el método "pd.read_csv" de la librería Pandas que lee archivos separados por comas para después obtener el nombre del archivo omitiendo la extensión y después se añade una nueva columna que asocia cada fila al tipo de archivo original. El siguiente paso es el reordenamiento de columnas para que la columna "label" aparezca siempre en el primer lugar, aunque esto lo modifique de forma manual en el Excel debido a una mejor visualización de los datos al momento de crear el archivo final. Una vez que se han procesado todos los datos se concatenan en un solo dataframe esto combina todas las filas en una tabla consolidada, reiniciando los índices. Los datos se agrupan por su etiqueta principal y el subcluster al que pertenecen. En cierto punto decidí que cada tipo de ropa debería de tener su propia etiqueta para poder tener un mejor procesamiento por lo que le asigne un único identificador, aunque en la versión de TMAP solo se utilizaron las columnas de categoría y subcluster. Finalmente, el archivo se almacena en la misma carpeta de entrada.

El código de "subcategorias_auto_fashion_mnist.py" es el encargado de la categorización automática de imágenes del conjunto de datos Fashion MNIST, el cual busca identificar subgrupos o subcategorías dentro de cada clase principal de ropa (camisetas, zapatos, bolsos, etc.).

Para lograrlo, se utilizan técnicas de reducción de dimensionalidad implementadas mediante las librerías de scikit-learn (sklearn). StandardScaler pertenece al módulo sklearn.preprocessing y se utiliza para escalar los datos numéricos. Su función es garantizar que todas las variables tengan la misma importancia durante los cálculos de distancia o varianza. En el caso de *Fashion MNIST*, las imágenes se representan como vectores con valores de píxeles (0–255). Sin escalamiento, los algoritmos de aprendizaje que se utilizan dentro de las librerías utilizadas podrían dar más peso a los valores más grandes, generando sesgos.

PCA (Principal Component Analysis) es una técnica de reducción de dimensionalidad que transforma un conjunto de variables correlacionadas en un conjunto más pequeño de componentes principales que retienen la mayor parte de la información.

Fashion MNIST contiene imágenes de 28x28 píxeles (784 características) lo que hace PCA es reducir esta dimensionalidad, por ejemplo, a 50 o 100 componentes, conservando los patrones visuales más importantes (formas, bordes, texturas, etc.).

KMeans es un algoritmo de clustering que busca dividir un conjunto de datos en K grupos (clusters), donde cada grupo contiene elementos similares entre sí y diferentes de los de otros grupos el cual se encuentra en el módulo sklearn.cluster.

En el flujo del código, después de aplicar PCA, se ejecuta K-Means para agrupar automáticamente los datos en subcategorías visualmente similares dentro de cada clase (label) del dataset.

Entorno en conda

Para la creación del entorno conda, instalamos las librerías desde tmap y utilizando a su vez conda-forge para los paquetes restantes y una vez teniendo el entorno creado, se realizó un archivo de tipo yaml, el cual se le llamo ‘environment.yml’, para que todos los miembros del equipo pudiéramos ejecutarlo.

Por último, se cambiaron los colores de los gradientes de cada subcluster para que fueran más sencillo diferenciarlos.

Resultado Final

En el análisis de los subclusters, se denota como se dividen en cuanto a la forma, sin embargo, había varios clusters juntos que no eran de la misma categoría ordenados en ramas iguales, pues tenían una forma extremadamente similar en la imagen y eso los agregaba en ramas que realmente no le corresponden. En cuanto a la división de clusters, estos si tienen sentido en si se observan las características individuales y observas la división, por ejemplo, una subdivisión entre zapatos deportivos y tacones señalizados en dos clusters diferentes.

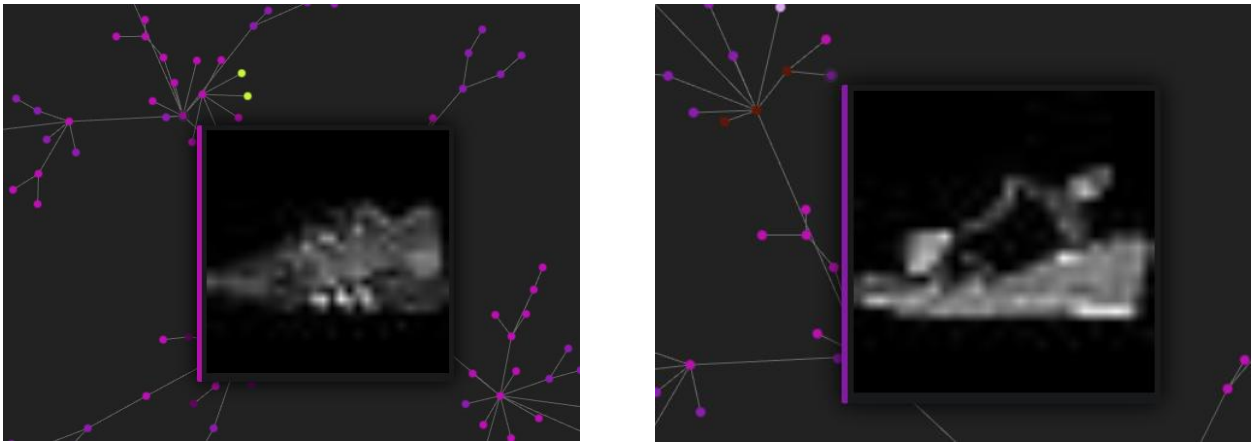

Fig. 1. Diferenciación entre clusters

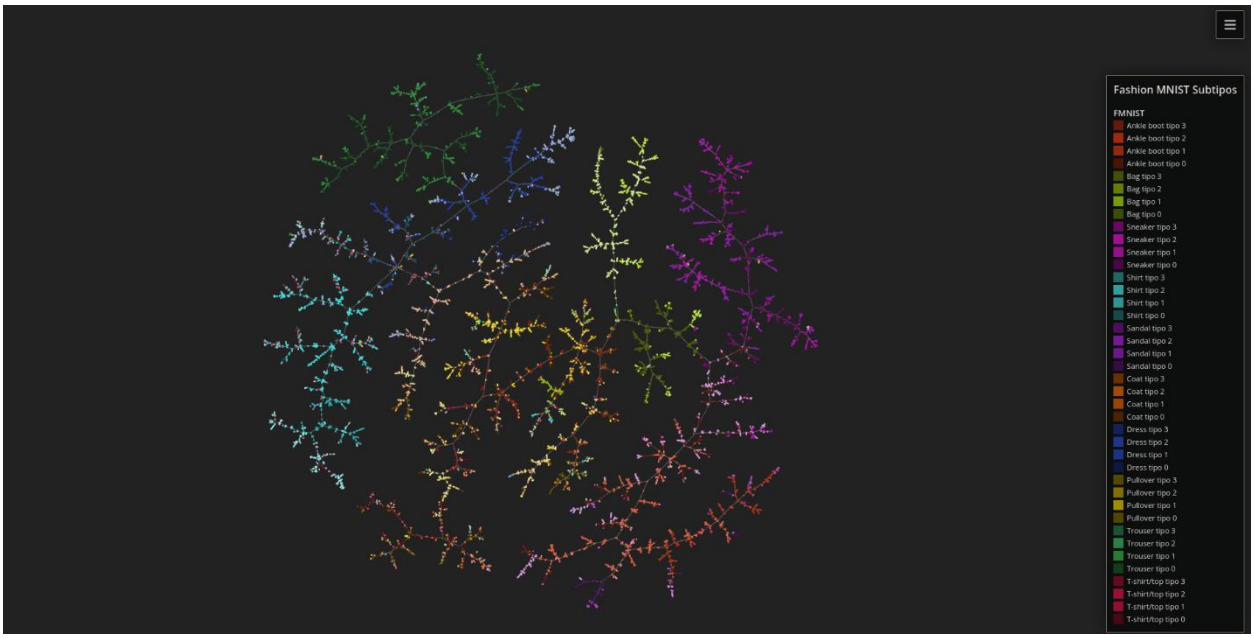

Fig. 2. Visualización de los datos con faerun y tmap.

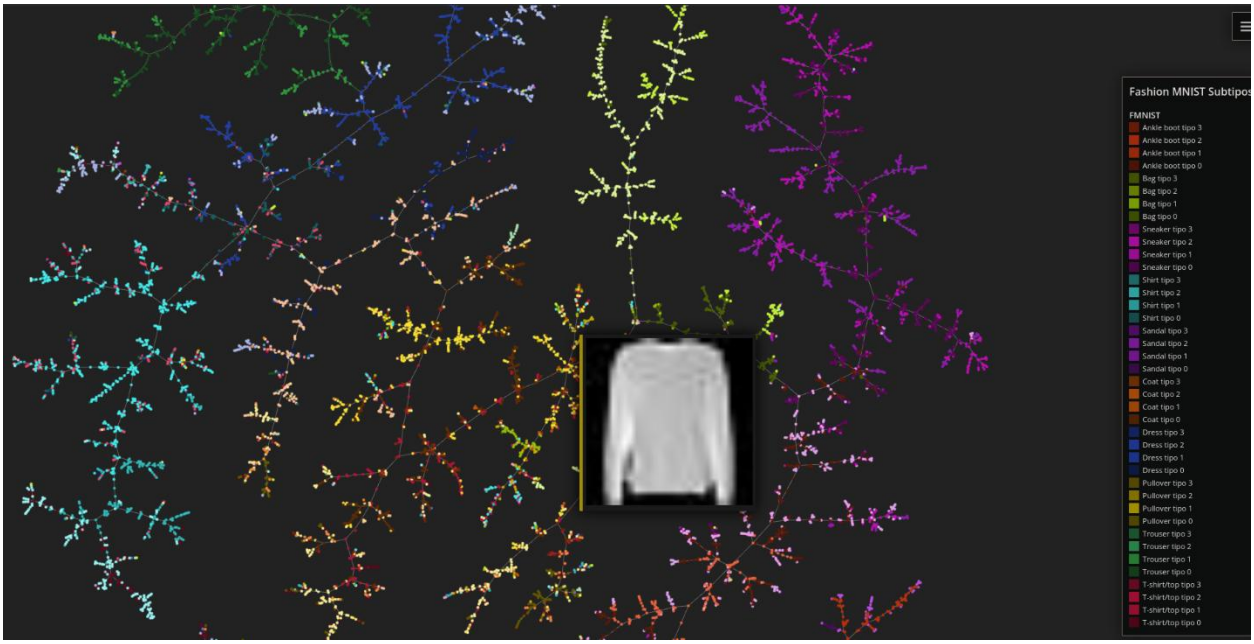

Fig. 3. Visualización de una prenda de ropa en el grafo.

El código se encuentra en el siguiente repositorio de GitHub para ser consultado: https://github.com/angel-lopez9177/Fashion-MNIST_Analisis-de-algoritmos/tree/develop

Conclusión

Se logro realizar la subclusterizacion de los datos del dataset Fashion Mnist, donde si tienen sentido las divisiones que realiza, sin embargo, el acomodo dentro de las ramas no es tan coherente como se esperaría, aunque mantiene cierta homogenización en cuanto a la generalidad de los datos manteniendo los colores en sus secciones correspondientes con algunos puntos de otros clusters alrededor de ellos. En general el resultado es bueno, pero se podría afinar más.

Referencias

Kaggle. (2025). Fashion MNIST [dataset]. Recuperado de https://www.kaggle.com/datasets/zalando-research/fashionmnist

tmap. (2025). *tmap – Visualize big high-dimensional data* [sitio web]. Recuperado de https://tmap.gdb.tools/

ZalandoResearch. (2025). *fashion-mnist* [repositorio en GitHub]. Recuperado de https://github.com/zalandoresearch/fashion-mnist