Agrupamiento de imágenes con mapas auto organizados

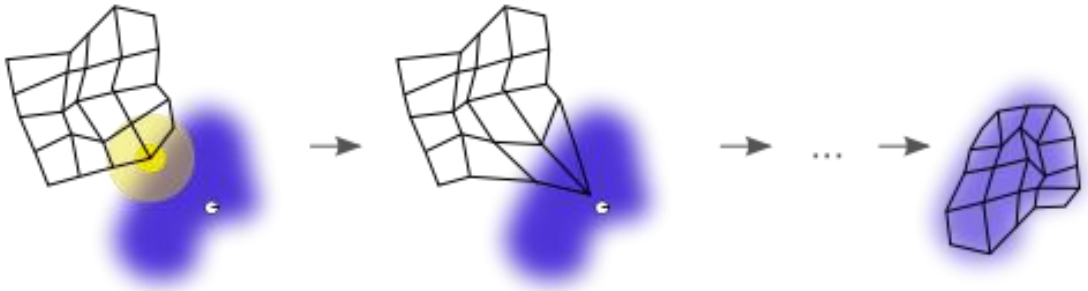


Imagen recuperada de https://es.wikipedia.org/wiki/Mapa_autoorganizado

El agrupamiento de imágenes mediante mapas autoorganizados es una técnica fascinante en el campo de la visión por computadora y el aprendizaje no supervisado. Este enfoque se basa en la idea de utilizar mapas autoorganizados, también conocidos como SOM (Self-Organizing Maps), para organizar y representar de manera eficiente la estructura subyacente de conjuntos de datos de imágenes.

Los mapas autoorganizados son una forma de red neuronal artificial que permite la proyección de datos de alta dimensión en una topología bidimensional o tridimensional, preservando las relaciones entre los datos originales. En el contexto del agrupamiento de imágenes, los SOM se entrenan para aprender representaciones espaciales de las características visuales presentes en las imágenes.

Durante el proceso de entrenamiento, los mapas autoorganizados asignan regiones específicas del mapa a grupos similares de imágenes, lo que facilita la identificación de patrones y la segmentación de datos. Este enfoque resulta particularmente útil en la organización y clasificación de grandes conjuntos de imágenes, ya que puede revelar estructuras subyacentes, relaciones y similitudes entre ellas sin requerir etiquetas previas.

- **¿Qué se hizo?**

En este trabajo se utilizará mapas autoorganizados para el agrupamiento de imágenes, utilizando tres diferentes conjuntos de datos de imágenes:

1. Fashion-MNIST: Esta es una base de datos de imágenes de la revistad Zalando que consiste en un set de entrenamiento de 60,000 ejemplos y un set de prueba de 10,000 ejemplos. Cada imagen es de 28 pixeles de altura por 28 pixeles de anchura, dando un total de 784 pixeles por imagen. Cada pixel tiene un único valor asociado, indicando la iluminación u oscuridad del pixel, donde un número alto indica un pixel más oscuro.

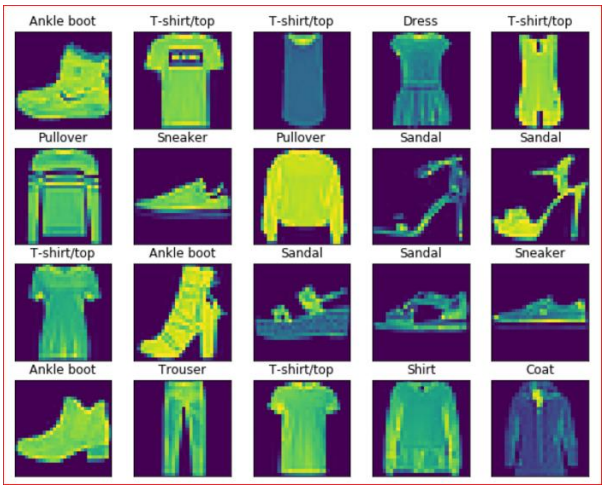
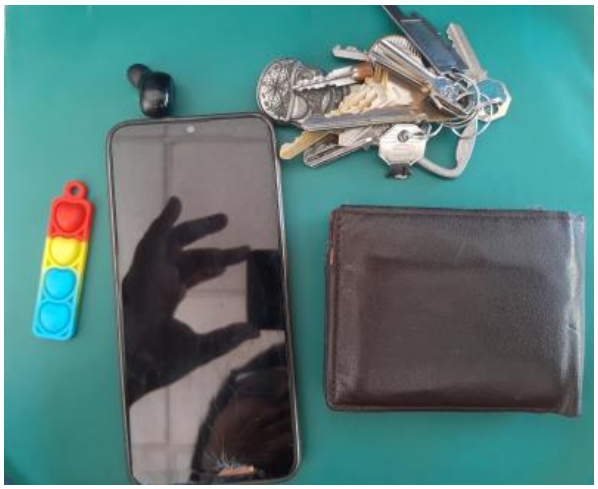


Imagen recuperada de https://medium.datadriveninvestor.com/implementing-convolutional-neural-network-using-tensorflow-for-fashion-mnist-caa99e423371

1. Fotografías satelitales: Esta base de datos tiene alrededor de 2000 fotografías satelitales de ambientes de México. Cada imagen está asociada a una de las siguientes clases: agua, bosque, ciudad, cultivo, desierto y montaña.

2. Imágenes de objetos: Nosotros, como equipo, creamos un conjunto de datos de imágenes de unas llaves, un celular, un audífono, un objeto antiestrés y una cartera. Todas estas fotos son de los mismos objetos con diferentes ángulos y niveles de iluminación para darle variedad computacional.



En el código base que se nos fue proporcionado para lograr obtener las características principales de nuestras imágenes para los 3 diferentes base de datos, primero, cargamos y redimensionamos las imágenes. Luego, se calculan los histogramas de color para los canales rojo, verde y azul, generando un descriptor de histograma de color normalizado. Posteriormente, se convierte la imagen a escala de grises y se calcula la matriz de co-ocurrencia de nivel de gris (GLCM) con ciertos parámetros. A partir de la GLCM, se extraen descriptores de textura, como disimilitud, homogeneidad, energía y correlación. Estos descriptores ofrecen información valiosa sobre las características visuales y texturales de cada una de nuestras imágenes, siendo útiles para nuestro modelo de clasificación de imágenes. El código se muestra a continuación:

```
import numpy as np

import matplotlib.pyplot as plt


from skimage import io

from skimage.transform import resize

from skimage.feature import graycomatrix, graycoprops

from skimage.color import rgb2gray
```

```
from skimage import img_as_ubyte


#--------------------------------------------------------------------------------
#   Load image
#--------------------------------------------------------------------------------
scale = 8
img_width = int(1920/scale)
img_height = int(1080/scale)
rgb = io.imread('img.jpg')
rgb_resized = resize(rgb, (img_height, img_width), anti_aliasing=True)


#--------------------------------------------------------------------------------
#   Color histograms
#--------------------------------------------------------------------------------
nbins = 16


rh = np.histogram(rgb_resized[:,:,0].flatten(), nbins, density = True)
gh = np.histogram(rgb_resized[:,:,1].flatten(), nbins, density = True)
bh = np.histogram(rgb_resized[:,:,2].flatten(), nbins, density = True)
hist_descriptor = np.concatenate((rh[0], gh[0], bh[0]))
```

```
#-------------------------------------------------------------------------------------------------------

#   Texture descriptors

#-------------------------------------------------------------------------------------------------------

gray_resized = img_as_ubyte(rgb2gray(rgb_resized))

glcm = graycomatrix(gray_resized, distances=[5], angles=[0, np.pi/4, np.pi/2, 3*np.pi/4])

texture_desc = [graycoprops(glcm, 'dissimilarity')[0, 0], graycoprops(glcm, 'homogeneity')[0, 0], graycoprops(glcm, 'energy')[0, 0], graycoprops(glcm, 'correlation')[0, 0]]
```

- **¿Dónde se hizo?**

IDE de Visual Studio es una plataforma de lanzamiento creativa que puede utilizar para editar, depurar y compilar código y, finalmente, publicar una aplicación. Además del editor y depurador estándar que ofrecen la mayoría de IDE, Visual Studio incluye compiladores, herramientas de completado de código, diseñadores gráficos y muchas más funciones para mejorar el proceso de desarrollo de software.

Imagen recuperada de https://mvpcluster.com/noticia/visual-studio/

- **¿Qué métodos se usaron para la clasificación de las imágenes?**

Existen diversas librerías en Python capaces de trabajar con mapas autoorganizados y representarlos gráficamente, tales como:

- **MiniSom:** Es una implementación minimalista basada en Numpy de los mapas autoorganizados (SOM, por sus siglas en inglés). SOM es un tipo de Red Neuronal Artificial capaz de convertir relaciones estadísticas complejas y no lineales entre elementos de datos de alta dimensión en relaciones geométricas simples en una visualización de baja dimensión. MiniSom está diseñado para permitir a los investigadores construir fácilmente sobre él y brindar a los estudiantes la capacidad de comprender rápidamente sus detalles.
- **Somoclu:** Es una implementación masivamente paralela de mapas autoorganizados. Depende de OpenMP para la ejecución en múltiples núcleos y puede ser acelerada por CUDA. La topología del mapa es ya sea plana o toroidal, y la rejilla es rectangular o hexagonal. Actualmente, este módulo de Python admite un subconjunto de la versión de línea de comandos.
- **NeuPy:** NeuPy es una biblioteca de Python para prototipar y construir redes neuronales. NeuPy utiliza Tensorflow como un backend computacional para modelos de aprendizaje profundo.
- **Kohonen:** La librería Kohonen en Python es una implementación simple y fácil de usar de mapas autoorganizados (SOM). Esta librería se centra en proporcionar herramientas para la creación y manipulación de mapas autoorganizados con una interfaz sencilla.
- **SOMPY:** Al utilizar SOMPY, los usuarios pueden aprovechar la potencia de los mapas autoorganizados para explorar la estructura subyacente y las relaciones en conjuntos de datos de manera visual y efectiva.

Como pudimos observar, existen librerías de sobra para escoger y trabajar con los mapas autoorganizados, pero en lo que a nuestro trabajo respecta, decidimos utilizar la librería MiniSom para mantener la simplicidad, ya que fue una librería que se nos recomendó desde un inicio.

Los mapas autoorganizados consisten en una red neuronal artificial para la agrupación y visualización de información. Este algoritmo descubre rasgos comunes y semejanzas en los datos de entrada, y agrupa las observaciones similares de manera automática y no supervisada. La red está compuesta por dos capas de neuronas; una de entrada y otra de salida. La capa de entrada consiste en vectores de pesos que reciben y transmiten a la capa de salida la información procedente del exterior. La

magnitud de cada vector corresponde al número de variables en el conjunto de datos de entrada y los pesos dependen de los valores de estas variables. La capa de salida está formada por m neuronas o unidades de mapa, que se encuentran organizadas, normalmente, en forma de una cuadrícula o mapa bidimensional.

La red SOM, como la mayoría de las redes neuronales artificiales, opera a través de dos etapas: entrenamiento y mapeo. En la etapa de entrenamiento se construye el mapa de salida en función de los ejemplos de entrada contenidos en un conjunto de datos de aprendizaje. Culminada la fase de entrenamiento, puede comenzar la fase de mapeo en la cual la red clasifica automáticamente a cada nuevo vector de entrada.

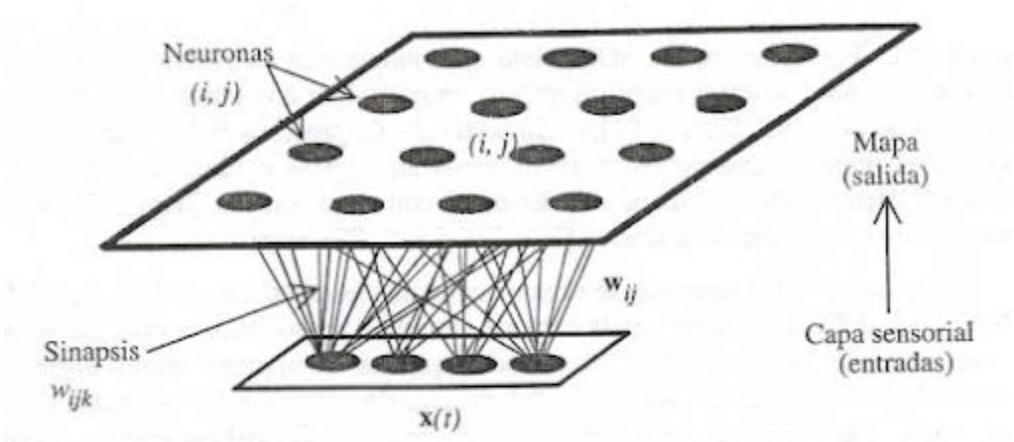


Imagen recuperada de https://www.researchgate.net/figure/Figura-214-Arquitectura-de-las-Redes-de-Kohonen-En-este-tipo-de-redes-el_fig4_49911738

El entrenamiento inicia asignando un vector modelo de pesos a cada unidad del mapa de salida en un espacio de datos multidimensional. Los valores iniciales de los pesos (w) son seleccionados aleatoriamente y cada individuo en los datos de entrada está representado por un vector con pesos determinados por los valores de las variables de entrada. El aprendizaje de los mapas autoorganizados es un procedimiento competitivo que compara distancias euclidianas entre los pesos de cada nuevo vector de entrada y los pesos de los vectores modelo de todas las unidades del mapa de salida. La unidad con la menor distancia euclidiana se selecciona como la mejor unidad de correspondencia o neurona ganadora. Este procedimiento actualiza los pesos no solo de la neurona ganadora, sino que también de las neuronas vecinas,

para preservar la topología de manera que los vectores que están cerca en el espacio de entrada se asignan a unidades que estén cerca en el mapa de salida.

De esta forma, la neurona ganadora se convierte en el centro de una vecindad de actualización de las unidades del mapa y sus pesos asociados, de manera que cada vector de peso converja con el patrón de entrada. La velocidad con la cual los nodos ganadores convergen hacia los vectores de entrada se denomina velocidad de aprendizaje.

A lo largo del aprendizaje, la velocidad de aprendizaje y el tamaño de la vecindad de actualización disminuyen, de modo que los patrones generalizados iniciales se refinan de manera progresiva.

Imagen recuperada de https://addi.ehu.es/bitstream/handle/10810/49091/TFG_Diaz_Rodriguez_Mikel.pdf?sequence=3

El objetivo principal del modelo es permitir que los pesos de los vectores de las unidades de salida aprendan de lo que se les presenta a los vectores de entrada. Este proceso es iterativo y se repite secuencialmente para cada nueva entrada hasta alcanzar el número de iteración predefinido para el aprendizaje.

Después de la dase de aprendizaje, la red SOM consta de varios vectores, con vectores similares cerca y vectores diferentes más separados.

- **¿Cómo se pueden evaluar los resultados?**

La evaluación del resultado de un Mapa Autoorganizado (SOM) implica comprender la calidad de la representación que el mapa ha aprendido sobre los datos y su capacidad para organizar y visualizar patrones subyacentes. Aquí hay algunas medidas y técnicas comunes para evaluar un SOM:

  - **Visualización del Mapa:** La visualización del propio mapa es una evaluación inicial y fundamental. Examina la distribución de las neuronas y cómo agrupan las diferentes regiones del espacio de entrada. Utiliza técnicas como la coloración de las neuronas para resaltar grupos o la representación de vectores de datos en el espacio del mapa.
  - **Distancias en el Mapa:** Calcula distancias en el mapa entre las neuronas y evalúa cómo reflejan las relaciones entre los datos en el espacio de entrada. Puedes usar medidas de distancia como la euclidiana o la distancia de Manhattan.
  - **Análisis de Vecindario:** Analiza cómo las neuronas vecinas en el mapa representan datos similares. Si dos neuronas están cerca en el mapa, sus respuestas deberían ser similares. Puedes explorar cómo cambian las respuestas de las neuronas en función de su proximidad en el mapa.
  - **U-Matrix (Matriz U):** La U-Matrix es una representación visual de las distancias entre las neuronas vecinas en el mapa. Muestra la conectividad y la coherencia del mapa, permitiendo identificar áreas densas y áreas menos densas en función de la distancia entre las neuronas.
  - **Histograma de Activación:** Examina la distribución de la actividad de las neuronas. Un histograma de activación puede revelar desequilibrios en la respuesta de las neuronas, indicando áreas del mapa con mayor o menor densidad de datos.
  - **Preservación Topológica:** Evalúa la preservación topológica del mapa, es decir, cómo se mantienen las relaciones espaciales entre los datos en el espacio de entrada y en el mapa. Puedes usar medidas como el índice de calidad topológica (TQI).
  - **Validación Externa:** Si dispones de etiquetas o información externa sobre tus datos, puedes realizar evaluaciones externas utilizando métricas de validación como la pureza o la precisión en la clasificación.
  - **Tiempo de Convergencia:** Observa el tiempo que lleva que el SOM converja durante el entrenamiento. Un SOM bien entrenado debería converger de manera eficiente.

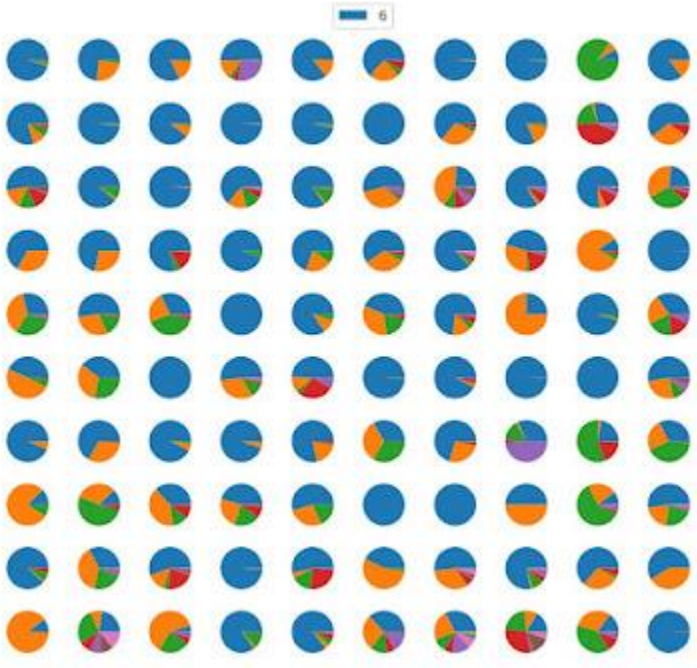
Es importante destacar que la elección de las métricas y técnicas de evaluación dependerá de la naturaleza específica de tus datos y del objetivo de tu análisis. No hay una métrica única que sirva para todos los casos, por lo que es recomendable utilizar múltiples enfoques para obtener una comprensión completa del rendimiento del SOM.

También, es necesario aclarar que no se utilizó ningún método de evaluación de resultados para este trabajo, pero en el caso de una posible réplica se dan a conocer todos estos métodos.

- **¿Qué resultados se obtuvieron?**

Debido a que en el código se específico que el mapa autoorganizado fuera de 10x10, los siguientes resultados se mostraran como gráficas, en la cual cada una presenta un clúster donde se clasificaron n imágenes, siendo n el número de imágenes de la base de datos sobre 100.

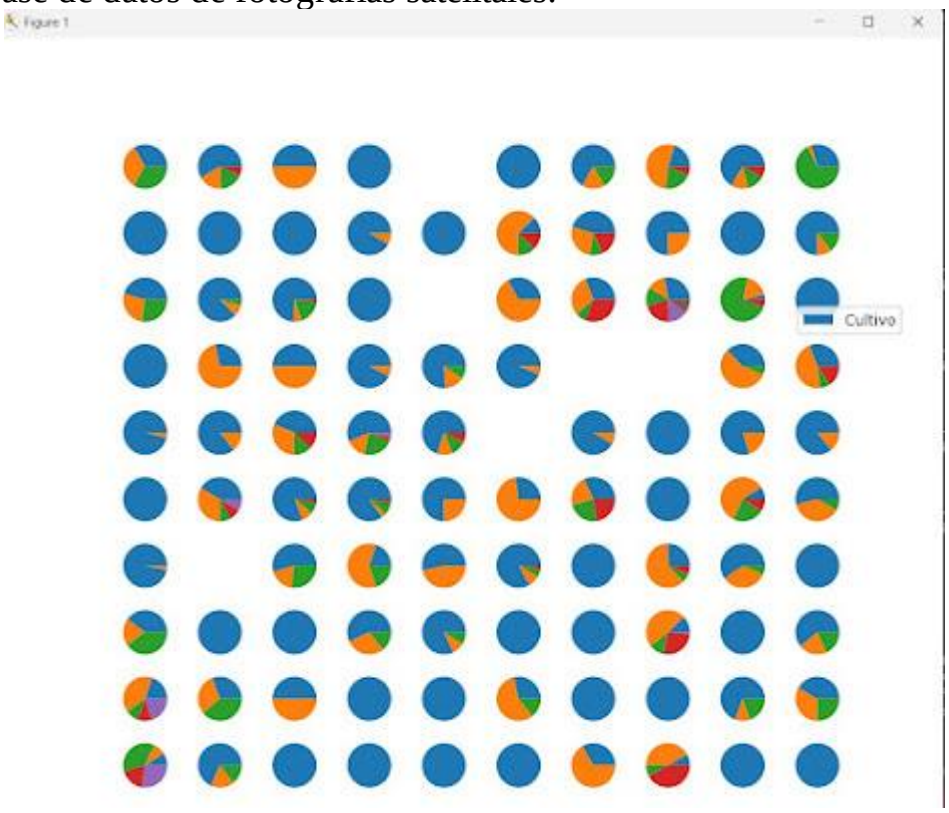
Con la base de datos Fashion-MNIST:



Cada gráfica de los resultados es una clasificación de 700 observaciones, ya que la base de datos cuenta con 70,000 imágenes.

Como podemos observar, el patrón más común en todo el mapa es que muchas imágenes fueron agrupadas como pertenecientes a la categoría azul, mientras que la segunda categoría más presente agrupada se puede apreciar como la categoría naranja.

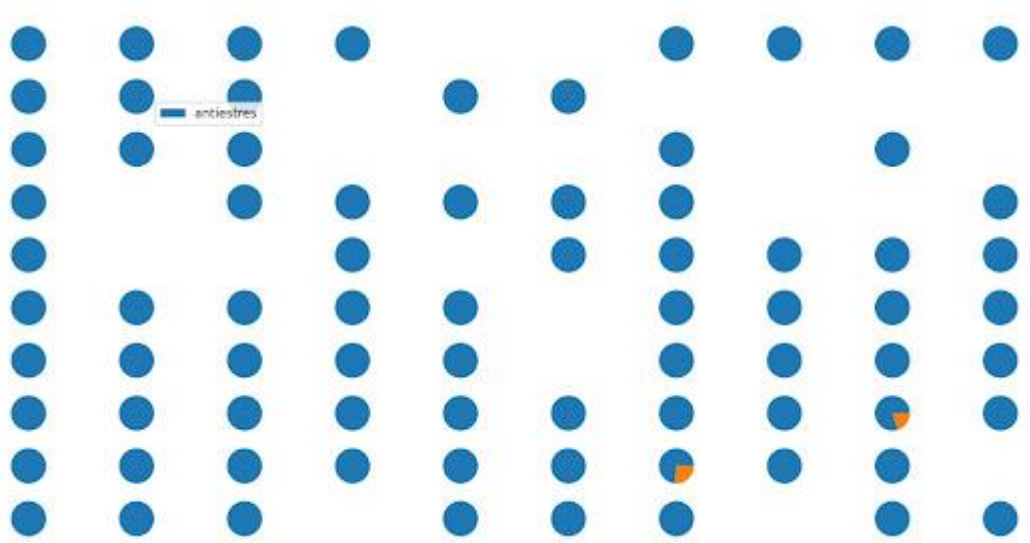
Con la base de datos de fotografías satelitales:



En esta base de datos, las gráficas cuentan con aproximadamente 20 observaciones cada una, esto debido a que la base de datos de imágenes satelitales tiene al rededor de 2000 tomas.

En este mapa autoorganizado podemos observar que la categoría más frecuente es la de cultivo, lo cual hace sentido, ya que de acuerdo a los resultados obtenidos mediante la clasificación de imágenes con modelos clásicos arrojaron el mismo resultado.

Con la base de datos de objetos:



En esta última base de datos hecha por nosotros, cada gráfica cuenta con 5 observaciones, puesto que el dataset consta de 500 imágenes de 5 objetos diferentes.

Y como podemos observar, la mayoría de estos pequeños clústeres han agrupado las observaciones como objetos anti estrés. Esto nos lleva a pensar que, si bien el modelo puede no estar distinguiendo de manera correcta las imágenes, podríamos haber ajustado de manera errónea el modelo.

- **Eficacia del modelo por dataset**
    - **Fashion-MNIST:** En este caso, se observa que el patrón más común en todo el mapa es la clasificación de muchas imágenes en la categoría azul, seguida por la categoría naranja. La consistencia en la clasificación de imágenes en estos dos clústeres puede sugerir que el modelo está capturando patrones distintivos en el conjunto de datos Fashion-MNIST. Sin embargo, sería necesario realizar una evaluación más detallada utilizando métricas específicas para la evaluación de agrupamientos, como la cohesión intracluster y la separación intercluster, para validar cuantitativamente la calidad del agrupamiento.
    - **Fotografías Satelitales:** En este contexto, la predominancia de la categoría "cultivo" en el mapa autoorganizado concuerda con los resultados obtenidos mediante la clasificación de imágenes con modelos clásicos. Este resultado refuerza la idea de que el SOM puede ser eficaz para capturar patrones relevantes en datos satelitales y proporcionar agrupamientos significativos. La baja cantidad de observaciones por gráfica puede afectar la generalización de estos resultados, por lo que sería beneficioso aumentar la cantidad de observaciones por gráfica o realizar una validación cruzada.

- **Base de Datos de Objetos:** Los resultados para la base de datos de objetos, donde la mayoría de los pequeños clústeres clasificaron las observaciones como "objetos anti estrés", plantea interrogantes sobre la calidad del modelo y la posibilidad de ajustes erróneos. Es importante considerar si el modelo ha sido sobreajustado a características específicas de ciertos objetos, lo que puede afectar negativamente la capacidad de generalización del modelo a nuevas instancias. En este caso, ajustar los parámetros del SOM o explorar otras técnicas de agrupamiento podría mejorar la eficacia del modelo.

La evaluación de modelos para agrupar datos no solo depende de la visualización del mapa, sino también de métricas cuantitativas que puedan medir la calidad de los agrupamientos. Además, ajustar el tamaño del mapa y otros parámetros del SOM puede influir significativamente en los resultados.

Aunque la visualización de los mapas autoorganizados proporciona una representación intuitiva de las agrupaciones, es esencial complementarla con evaluaciones cuantitativas y considerar ajustes en los parámetros del modelo para mejorar su rendimiento en diferentes conjuntos de datos.

- **Conclusión**

En conclusión, el trabajo ha explorado la aplicación de mapas autoorganizados (SOM) para el agrupamiento de imágenes en tres conjuntos de datos diversos: Fashion-MNIST, Fotografías Satelitales y Objetos creados por el equipo. A través de la utilización de la librería MiniSom y la visualización de los resultados en mapas 10x10, se ha proporcionado una visión general de la eficacia del modelo en la organización y clasificación de diferentes tipos de imágenes.

Sin embargo, la evaluación de la base de datos de objetos revela desafíos potenciales. Esto plantea preguntas sobre la adecuación del modelo para este conjunto de datos específico y sugiere posibles ajustes erróneos.

Aunque no se han utilizado métricas de evaluación cuantitativa en este trabajo, la discusión sobre la eficacia de los modelos destaca la importancia de complementar la visualización del mapa con medidas objetivas, como distancias en el mapa, análisis de vecindario y preservación topológica, para una comprensión más profunda del rendimiento del SOM.

Referencias:

1. **Fashion MNIST. (2017, 7 diciembre). Kaggle. https://www.kaggle.com/datasets/zalando-research/fashionmnist**
2. **Visual Studio: IDE y editor de código para desarrolladores de software y teams. (2023, 27 noviembre). Visual Studio. https://visualstudio.microsoft.com/es/#vs-section**
3. **Introduction — Somoclu 1.7.5 documentation. (s. f.). https://somoclu.readthedocs.io/en/stable/**
4. **JustGlowing. (s. f.). GitHub - JustGlowing/minisom: :red_circle: MiniSom is a minimalistic implementation of the self organizing maps. GitHub. https://github.com/JustGlowing/minisom**
5. **NeuPy — NeuPy. (s. f.). http://neupy.com/pages/home.html**
6. **Sevamoo. (s. f.). GitHub - Sevamoo/SOMPY: A Python Library for Self Organizing Map (SOM). GitHub. https://github.com/sevamoo/SOMPY**
7. **Valera, Á. R. (2019). AGRUPAMIENTO DE SUELOS CON REDES NEURONALES DE MAPAS AUTOORGANIZADOS EN PAISAJES DE MONTAÑA EN LA REGIÓN CENTRO NORTE DE VENEZUELA. Redalyc.org. https://www.redalyc.org/articulo.oa?id=72164777005**