Clasificación de imágenes con métodos clásicos y redes neuronales

Imagen recuperada de https://inteligenciaartificials.com/investigacion-sobre-redes-neuronales-en-ia/

La clasificación de imágenes es una tarea fundamental en el campo de la visión por computadora, y las redes neuronales han revolucionado significativamente esta área en las últimas décadas. Las redes neuronales son modelos computacionales inspirados en el funcionamiento del cerebro humano, y su capacidad para aprender patrones complejos las hace ideales para abordar problemas de clasificación de imágenes.

El proceso de entrenamiento de una red neuronal implica presentarle un conjunto de imágenes etiquetadas, permitiendo que la red ajuste sus parámetros para minimizar la diferencia entre las predicciones y las etiquetas reales. Este enfoque basado en el aprendizaje supervisado ha demostrado ser efectivo en la capacitación de modelos para reconocer patrones en una amplia variedad de imágenes.

Con el advenimiento de conjuntos de datos masivos y avances en hardware de alto rendimiento, las redes neuronales han alcanzado niveles de precisión impresionantes en la clasificación de imágenes, superando a los métodos tradicionales que hemos de observar en este trabajo.

- **¿Qué se hizo?**

Se evaluaron arquitecturas clásicas de redes neuronales para el manejo de imágenes utilizando los siguientes conjuntos de datos:

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.

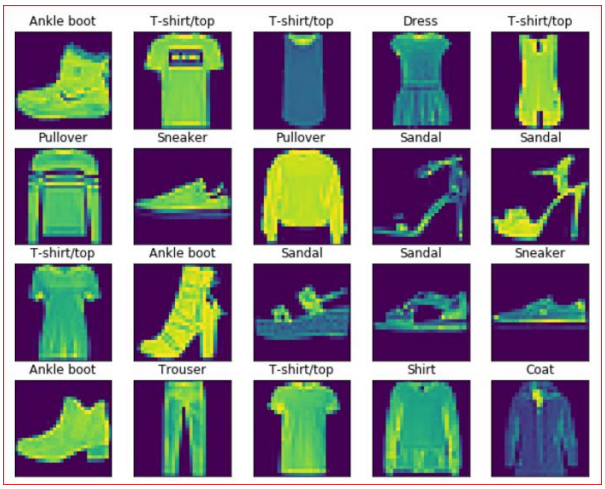
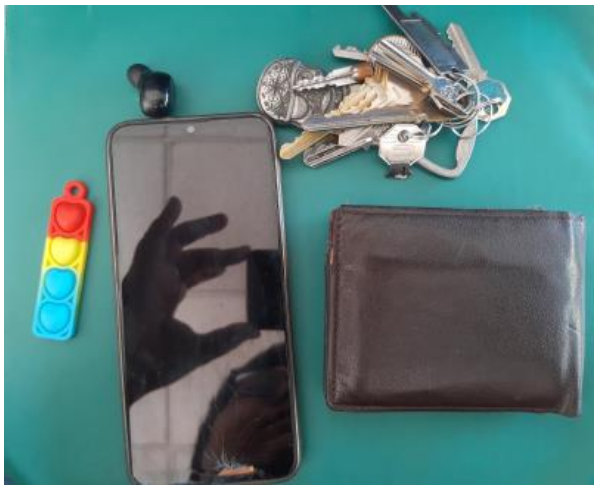


Imagen recuperada de https://medium.datadriveninvestor.com/implementing-convolutional-neural-network-using-tensorflow-for-fashion-mnist-caa99e423371

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.



Para trabajar con dichas imágenes de las bases de datos, se extrajeron características como el color y la textura de las imágenes con el siguiente código de Python:

```
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?**

Para evaluar las arquitecturas de redes neuronales se utilizo la plataforma de oneAPI, una especificación abierta y basada en estándares, soportando múltiples tipos de arquitecturas que no se limitan al GPU (Unidad de Procesamiento Gráfico), CPU (Unidad Central de Procesamiento) y FPGA (Matriz de Puerta Programada en Campo). OneAPI tiene ambas; una API (Interfaz de Programación de Aplicaciones) basada en paradigmas de la programación y una programación directa.

Imagen recuperada de https://buy.hpe.com/lamerica/es/software/third-party-software/third-party-software/third-party-software/licencia-electr%C3%B3nica-de-uso-de-3-a%C3%B1os-de-intel-oneapi-base-y-soporte-limitado-de-computaci%C3%B3n-de-alto-rendimiento-para-un-m%C3%A1ximo-de-64-nodos-de-hardware-distinto-a-intel-para-hasta-10-usuarios/p/r9p29a

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

Se utilizaron dos modelos de aprendizaje supervisado y un modelo de perceptrón multicapa. Siendo un modelo lineal, uno no lineal y otro de aprendizaje profundo. Estos métodos fueron:

  - **SVM (Máquina de Soporte Vectorial):** Support Vector Machine es un algoritmo de aprendizaje supervisado que se utiliza en problemas de clasificación y regresión. Su objetivo es encontrar un hiperplano que separe de la mejor forma posible dos clases diferentes de puntos de datos. Esto implica el hiperplano con el margen más amplio entre las dos
    clases.
    Este algoritmo pertenece a una clase de algoritmos de Machine Learning denominados métodos kernel, donde se puede utilizar una función de kernel para transformar las características. Las funciones de kernel asignan los datos a un espacio dimensional diferente, con la expectativa de que resulte más fácil separar las clases después de esta transformación, simplificando los límites de decisión complejos no lineales para hacerlos lineales en el espacio dimensional de características superior asignado. A esto se le conoce como truco de kernel.
    - **SVM lineal:** El kernel lineal busca encontrar un hiperplano óptimo que maximice el margen entre las clases en el espacio de características cuantificando la similitud de un par de observaciones

usando la correlación de Pearson. El coeficiente de correlación de Pearson es una prueba que mide la relación estadística entre dos variables continuas.

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

Imagen recuperada de https://rpubs.com/Cristina_Gil/SVM#:~:text=Tambi%C3%A9n%20puede%20definirse%20como%20una,en%20un%20nuevo%20espacio%20dimensional.&text=El%20kernel%20lineal%20cuantifica%20la,usando%20la%20correlaci%C3%B3n%20de%20Pearson.

- **SVM de Base Radial:** El kernel radial tiene un comportamiento local, en el sentido de que solo las observaciones del set de entrenamiento cercanas a una observación del set de prueba tendrán efecto sobre la clasificación de las observaciones. Es una extensión del SVM lineal que utiliza una función de kernel radial para permitir la clasificación efectiva de conjuntos de datos no linealmente separables. La elección adecuada del parámetro γ es crucial para el rendimiento del modelo. El corazón del SVM de base radial es la función de kernel radial, también conocida como la función de base radial. La función de kernel RBF se define como

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$

Imagen recuperada de https://rpubs.com/Cristina_Gil/SVM#:~:text=Tambi%C3%A9n%20puede%20definirse%20como%20una,en%20un%20nuevo%20espacio%20dimensional.&text=El%20kernel%20lineal%20cuantifica%20la,usando%20la%20correlaci%C3%B3n%20de%20Pearson.

  - donde "x" y "x'" son dos puntos en el espacio de características original, y γ es un parámetro que controla la forma de la función de kernel. La función de kernel RBF asigna pesos a los puntos de datos en función de su distancia alrededor de un punto central, permitiendo capturar patrones no lineales.

- **Perceptrón Multicapa:** Es un algoritmo tipo red neuronal artificial organizada en capas que recibe variables y en función de unos parámetros entrega un resultado que es la clasificación de un objeto en función de las variables propias del objeto. Este modelo se entrena con el objetivo de identificar los valores óptimos para los parámetros libres que permitan clasificar lo mejor posible. Los "parámetros libres" del modelo son el número de capas de la red, la función de activación, la función de costo y el learning rate.

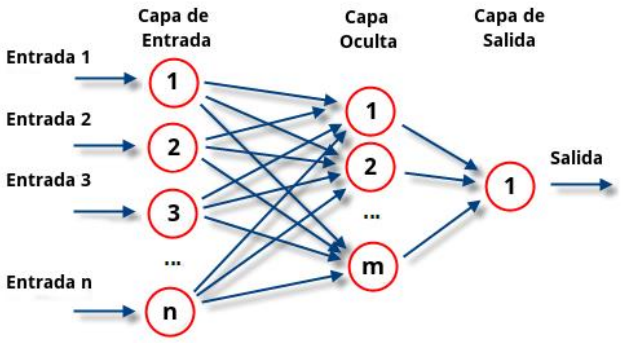


Imagen recuperada de https://es.wikipedia.org/wiki/Perceptr%C3%B3n_multicapa

- **¿Cómo se evaluaron los resultados?**

Para evaluar el rendimiento de modelos de clasificación, se utilizan diversas métricas que proporcionan información sobre la calidad de las predicciones. Para este caso se utilizaron las siguientes métricas:

  - **Exactitud (Accuracy):**
    - Es la proporción de predicciones correctas respecto al total de predicciones.
    - La exactitud es fácil de entender, pero puede ser engañosa en conjuntos de datos desbalanceados.
  - **Recall:**
    - Mide la proporción de instancias positivas correctamente clasificadas respecto al total de instancias clasificadas como positivas.
  - **Cross Validation (Validación Cruzada):**
    - Es una técnica utilizada en el campo del aprendizaje automático para evaluar el rendimiento de un modelo y reducir el riesgo de sobreajuste. Su propósito es obtener una estimación más confiable del rendimiento del modelo al entrenar y evaluar el modelo con múltiples subconjuntos de datos. Este método es especialmente útil cuando el tamaño del conjunto de datos es limitado y se quiere maximizar el uso de los datos disponibles para el entrenamiento y la evaluación.

- **Resultados**

Los resultados obtenidos con los 3 modelos de clasificación son los siguientes:

  - **Base de datos Fashion-MNIST**
    - **SVM lineal:**

Linear SVM Classification Report:

	precision	recall	f1-score	support
0	0.75	0.83	0.79	1394
1	0.96	0.97	0.96	1402
2	0.74	0.77	0.75	1407
3	0.85	0.84	0.85	1449
4	0.76	0.77	0.76	1357
5	0.93	0.94	0.94	1449
6	0.67	0.58	0.62	1407
7	0.91	0.93	0.92	1359
8	0.95	0.90	0.93	1342
9	0.96	0.93	0.94	1434
accuracy			0.85	14000
macro avg	0.85	0.85	0.85	14000
weighted avg	0.85	0.85	0.85	14000

- **SVM de base radial:**

SVM RBF Classification Report:

	precision	recall	f1-score	support
0	0.83	0.86	0.84	1394
1	1.00	0.97	0.98	1402
2	0.83	0.82	0.83	1407
3	0.88	0.91	0.89	1449
4	0.81	0.84	0.82	1357
5	0.97	0.96	0.96	1449
6	0.74	0.66	0.70	1407
7	0.93	0.96	0.95	1359
8	0.94	0.98	0.96	1342
9	0.97	0.95	0.96	1434
accuracy			0.89	14000
macro avg	0.89	0.89	0.89	14000
weighted avg	0.89	0.89	0.89	14000

- **Perceptrón multicapa:**

MLP Classification Report:

	precision	recall	f1-score	support
0	0.84	0.82	0.83	1394
1	0.98	0.98	0.98	1402
2	0.76	0.85	0.80	1407
3	0.89	0.89	0.89	1449
4	0.82	0.76	0.79	1357
5	0.97	0.95	0.96	1449
6	0.71	0.69	0.70	1407
7	0.93	0.97	0.95	1359
8	0.97	0.97	0.97	1342
9	0.97	0.95	0.96	1434
accuracy			0.88	14000
macro avg	0.88	0.88	0.88	14000
weighted avg	0.88	0.88	0.88	14000

- **Base de datos de biomas**
  - **SVM lineal:**

```
Linear SVM Classification Report:
              precision    recall  f1-score   support

       Agua       0.83      0.87      0.85        68
     Bosque       0.85      0.90      0.87        68
     Ciudad       0.75      0.97      0.84        61
    Cultivo       0.88      0.67      0.76        76
   Desierto       1.00      0.85      0.92        68
    Montaña       0.77      0.81      0.79        63

   accuracy                           0.84       404
  macro avg       0.85      0.84      0.84       404
weighted avg       0.85      0.84      0.84       404
```

- **SVM de base radial:**

```
SVM RBF Classification Report:
              precision    recall  f1-score   support

       Agua       0.89      0.87      0.88        68
     Bosque       0.86      0.88      0.87        68
     Ciudad       0.73      0.90      0.81        61
    Cultivo       0.90      0.57      0.69        76
   Desierto       0.91      0.85      0.88        68
    Montaña       0.70      0.90      0.79        63

   accuracy                           0.82       404
  macro avg       0.83      0.83      0.82       404
weighted avg       0.84      0.82      0.82       404
```

- **Perceptrón multicapa:**

```
MLP Classification Report:
              precision    recall  f1-score   support

       Agua       0.89      0.87      0.88        68
     Bosque       0.86      0.90      0.88        68
     Ciudad       0.84      0.95      0.89        61
    Cultivo       0.90      0.79      0.84        76
   Desierto       0.97      0.87      0.91        68
    Montaña       0.76      0.84      0.80        63

   accuracy                           0.87       404
  macro avg       0.87      0.87      0.87       404
weighted avg       0.87      0.87      0.87       404
```

- **Resultados de Cross Validation:**

```
Linear SVM Cross-Validation Scores: [0.83415842 0.7617866  0.80397022 0.67493797 0.79404467]
Mean Linear SVM Cross-Validation Score: 0.7737795739871755
RBF SVM Cross-Validation Scores: [0.84158416 0.73449132 0.82630273 0.68238213 0.78163772]
Mean RBF SVM Cross-Validation Score: 0.7732796108394959
MLP Cross-Validation Scores: [0.81188119 0.86352357 0.9057072  0.88833747 0.88585608]
Mean MLP Cross-Validation Score: 0.8710611011473356
```

- **Base de datos de objetos**
  - **SVM lineal:**

```
Linear SVM Classification Report:
              precision    recall  f1-score   support

  antiestres       1.00      1.00      1.00       100
    audifono       1.00      1.00      1.00        88
     cartera       1.00      1.00      1.00       108
     celular       1.00      1.00      1.00       111
      llaves       1.00      1.00      1.00        93

    accuracy                           1.00       500
   macro avg       1.00      1.00      1.00       500
weighted avg       1.00      1.00      1.00       500
```

  - **SVM de base radial:**

```
SVM RBF Classification Report:
              precision    recall  f1-score   support

  antiestres       1.00      1.00      1.00       100
    audifono       1.00      1.00      1.00        88
     cartera       1.00      1.00      1.00       108
     celular       1.00      1.00      1.00       111
      llaves       1.00      1.00      1.00        93

    accuracy                           1.00       500
   macro avg       1.00      1.00      1.00       500
weighted avg       1.00      1.00      1.00       500
```

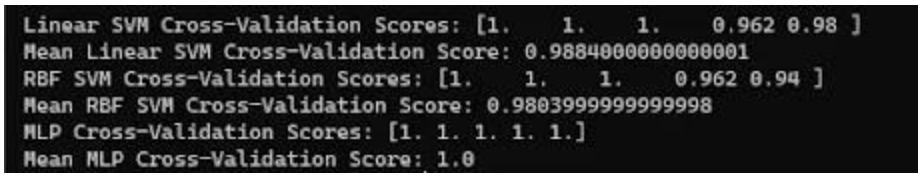
  - **Perceptrón multicapa:**

```
MLP Classification Report:
              precision    recall  f1-score   support

  antiestres       1.00      1.00      1.00       100
    audifono       1.00      1.00      1.00        88
     cartera       1.00      1.00      1.00       108
     celular       1.00      1.00      1.00       111
      llaves       1.00      1.00      1.00        93

    accuracy                           1.00       500
   macro avg       1.00      1.00      1.00       500
weighted avg       1.00      1.00      1.00       500
```

  - **Resultados de Cross Validation:**



A partir de los resultados puede observarse el desempeño de los 3 modelos entrenados con las diferentes bases de datos, y un común que comparten es que el modelo de perceptrón multicapa cuenta con los mejores resultados en las métricas de evaluación de casi todas las bases de datos.

En la base de datos Fashion-MNIST observamos exactitudes mayores a 0.8 en todos los modelos, y algunas variaciones en los recalls. El modelo con mejor exactitud fue la máquina de soporte vectorial de base radial, seguido del perceptrón multicapa y finalmente la máquina de soporte vectorial lineal. No se realizó validación cruzada para esta base de datos debido al tamaño de la misma.

La clase mejor clasificada fue la 1, y la que peor fue clasificada fue la clase 6.

En la base de datos de biomas encontramos muy buenos resultados por parte del perceptrón multicapa en la exactitud, recall de las clases y la validación cruzada, siendo el modelo que mejor clasificó dicho conjunto de datos. Las máquinas de soporte vectorial lineal y de base radial también lograron un buen desempeño, y podríamos atribuirle el que no fueran mejores al perceptrón el hecho de que las imágenes de esta base de datos demuestran una complejidad mayor a las de la base de datos Fashion-MNIST, debido a los diferentes colores y variaciones de los mismos. El perceptrón resulta útil para imágenes complejas, al menos más útil que modelos clásicos.

El bioma mejor clasificado fue el de ciudad, y el peor fue el bioma de cultivo.

Finalmente, en la base de datos de objetos encontramos un sobreajuste en todos los modelos, ya que todos cuentan con una exactitud y recall por clase perfectos, así como una validación cruzada casi perfecta. Esto nos lleva a pensar que las imágenes no mostraban mucha variedad entre si mismas, a excepción de si el objeto mostrado en la imagen era diferente. Es decir, las diferentes fotos tomadas a los objetos en cuestión eran muy parecidas entre sí y fueron clasificadas perfectamente.

- **Conclusión**

La elección entre SVM y MLP para clasificación de imágenes depende de varios factores, como el tamaño del conjunto de datos, la complejidad del problema y los recursos computacionales disponibles.

Para conjuntos de datos pequeños o problemas donde las relaciones son no lineales y complejas, los MLP pueden ser más adecuados.

Los SVM, especialmente con trucos de kernel, pueden ser eficaces cuando las características son altamente dimensionales o cuando se dispone de un conjunto de datos limitado.

En la práctica, puede ser beneficioso experimentar con ambos modelos, ajustar los hiperparámetros de manera adecuada y utilizar técnicas como la validación cruzada para evaluar su rendimiento en el conjunto de datos específico.

Referencias:

1. **Fashion MNIST. (2017, 7 diciembre). Kaggle. https://www.kaggle.com/datasets/zalando-research/fashionmnist**
2. **Mongkolsmai, T. (2022, 9 septiembre).** ***What is oneAPI: Demystifying oneAPI for Developers*****. Intel. https://www.intel.com/content/www/us/en/developer/articles/technical/oneapi-what-is-it.html#gs.0sh0if**
3. **Support Vector Machine (SVM). (s. f.). MATLAB & Simulink. https://la.mathworks.com/discovery/support-vector-machine.html**
4. **RPUBs - Máquinas de vector soporte. (s. f.). https://rpubs.com/Cristina_Gil/SVM#:~:text=Tambi%C3%A9n%20puede%20definirse%20como%20una,en%20un%20nuevo%20espacio%20dimensional.&text=El%20kernel%20lineal%20cuantifica%20la,usando%20la%20correlaci%C3%B3n%20de%20Pearson.**
5. **Ortega, C. (2023, 23 febrero). ¿Qué es el coeficiente de correlación de Pearson? QuestionPro. https://www.questionpro.com/blog/es/coeficiente-de-correlacion-de-pearson/**
6. **Buitrago, B. (2021, 16 diciembre). Redes neuronales — Perceptrón Multicapa i - iWannaBeDataDriven - Medium. Medium. https://medium.com/iwannabedatadriven/redes-neuronales-perceptr%C3%B3n-multicapa-i-d8c05a88857e**
7. **Validación cruzada - Amazon Machine Learning. (s. f.). https://docs.aws.amazon.com/es_es/machine-learning/latest/dg/cross-validation.html**