TECNOLÓGICO NACIONAL DE MEXICO

INSTITUTO TECNOLÓGICO DE LA LAGUNA

Inteligencia artificial

INGENIERÍA EN SISTEMAS COMPUTACIONALES

DOCENTE: LAMIA HAMDÁN M.

PROYECTO UNIDAD 4

NÚM DE CONTROL	NOMBRE
19130527	Fatima Gorety Garcia Yescas
19130519	Roberto Esquivel Troncoso
19130535	Ivan Herrera Garcia
19130514	Isaias Gerardo Cordova Palomares
19130547	Jesus Rafael Medina Dimas

FECHA DE ENTREGA: 16/10/2022

Índice

Introducción

Para el desarrollo de los programas de esta unidad utilizamos nuestros conocimientos en python para poder aplicar una red neuronal de perceptrón múltiple y la red neuronal para reconocimiento de figuras en imágenes, se utilizaron librerías como pandas, numpy, opencv, matplotlib e imultis.

Se nos mostrará un programa utilizando la red neuronal ART para reconocer figuras dentro de las imágenes, identificándo las y marcándose con un relieve.

Red neuronal ART para el reconocimiento de figuras o imágenes.

La teoría de la resonancia adaptativa (en inglés, adaptive resonance theory, conocido por sus siglas inglesas ART), desarrollada por Stephen Grossberg y Gail Carpenter. Es un modelo de red neuronal artificial (RNA) que basa su funcionamiento en la manera en que el cerebro procesa información y que describe una serie de modelos de redes neuronales que utilizando métodos de aprendizaje supervisado y no supervisado abordan problemas tales como el reconocimiento y la predicción de patrones.

El modelo ART soluciona el dilema de la estabilidad y plasticidad del aprendizaje mediante un mecanismo de realimentación entre las neuronas competitivas de la capa de salida.
Cuando a la red se le presenta un patrón de entrada este se hace resonar con los prototipos de las categorías conocidas por la red, si el patrón entra en resonancia con alguna clase entonces es asociado a esta y el centro de cluster es desplazado ligeramente para adaptarse mejor al nuevo patrón que le ha sido asignado. En caso contrario, si el patrón no entra en resonancia con ninguna clase, pueden suceder dos cosas: si la red posee una capa de salida estática entrará en saturación pues no puede crear una nueva clase para el patrón presentado pero tampoco puede asignarlo a una clase existente, si la red posee una capa de salida dinámica se creará una nueva clase para dicho patrón, esto no afectará a las clases ya existentes.

Librerías utilizadas

Dentro del proyecto de la red neuronal ART para el reconocimiento de figuras se utilizaron 4 librerías, estas se importaron al inicio de la codificación:

- import cv2
- import numpy as np
- from matplotlib import pyplot as plt
- import imutils

OpenCV

Esta es una enorme biblioteca de código abierto para visión por computadora, aprendizaje automático y procesamiento de imágenes. Puede procesar imágenes y videos para identificar objetos, rostros o incluso la escritura a mano de un ser humano. Cuando se integra con varias bibliotecas, como la que es una biblioteca altamente optimizada para operaciones numéricas, entonces el número de armas aumenta en su Arsenal, es decir, cualquier operación que uno pueda hacer en Numpy se puede combinar con OpenCV.

El método carga una imagen del archivo especificado. Si la imagen no se puede leer (debido a la falta de archivo, permisos incorrectos, formato no compatible o no válido), este método devuelve una matriz vacía.cv2.imread()

Sintaxis: cv2.imread(ruta, indicador)
Parámetros:
path: Cadena que representa la ruta de la imagen que se va a leer.
flag: Especifica la forma en que se debe leer la imagen. Su valor predeterminado es cv2. IMREAD_COLOR
Valor devuelto: Este método devuelve una imagen que se carga desde el archivo especificado.

Librería Numpy

NumPy es una librería de Python especializada en el cálculo numérico y el análisis de datos, especialmente para un gran volumen de datos.
Incorpora una nueva clase de objetos llamados arrays que permite representar colecciones de datos de un mismo tipo en varias dimensiones, y funciones muy eficientes para su manipulación.
La ventaja de Numpy frente a las listas predefinidas en Python es que el procesamiento de los arrays se realiza mucho más rápido (hasta 50 veces más) que las listas, lo cual la hace ideal para el procesamiento de vectores y matrices de grandes dimensiones.

La librería Matplotlib

Matplotlib es una librería de Python especializada en la creación de gráficos en dos dimensiones.

La librería Imutils

Una serie de funciones de conveniencia para facilitar las funciones básicas de procesamiento de imágenes, como la traducción, la rotación, el cambio de tamaño, la esqueletización y la visualización de imágenes matplotlib con OpenCV y Python 2.7 y Python 3.

Descripción del proyecto

Este programa funciona con la librería cv2 de openCV, en conjunto con numpy, matplotlib inútils, se puede analizar una imagen y obtener las formas geométricas que contiene (cuadrados, rectángulos, círculos, triángulos, entre otros), además usa opencv para el escaneo de la imagen ya que cambia el formato de la misma a una representación binaria, a esta representación binaria se le aplican diferentes métodos como el THRESH BINARY y threshold, y con opencv se visualiza la misma imagen en un frame con las figuras identificadas y resaltadas.

En términos generales lo que hace es modificar la imagen, haciéndola gris y quitándole la iluminación, para posteriormente buscar el número de vértices de cada forma y las revisa en las librerías y si los encuentra, marca el contorno.

Implementación de código

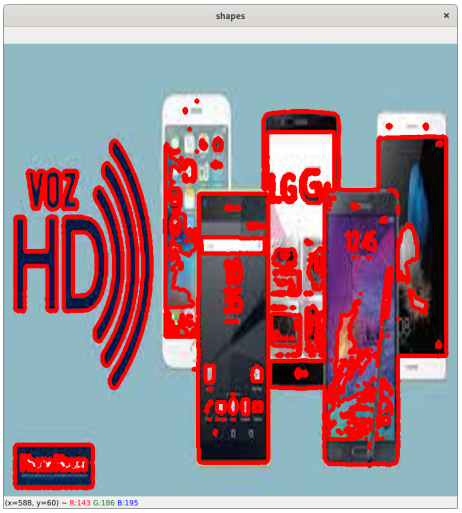
```
#----------------Librerías  que se usan----------------------
import cv2
import numpy as np
from matplotlib import pyplot as plt
import imutils
#----------------Fin librerías que se usan----------------------

#---------------Ejemplos de imágenes utilizadas-----------------
# Se carga la imagen, ya sea figuras o una imagen
# dando la ruta de la imagen
img2 = cv2.imread('/home/Bellkhen/Descargas/opencv-shape-detection/shapes.png')

#Imágenes de prueba  #Líneas comentadas
#img2=cv2.imread('/home/Bellkhen/Descargas/opencv-shape-detection/shapes.jpg')
#img2 = cv2.imread("/home/Bellkhen/Descargas/otrost.jpg")
#img2 = cv2.imread("/home/Bellkhen/Descargas/mona-china.jpg")
#----------------Fin ejemplo de imágenes utilizadas--------------

# Se renderiza la imagen a un tamaño dado (750 * 750), por si el tamaño de la
# imagen es pequeña o muy grande
img = cv2.resize(img2, (750,750))

# Se cambia el color de la imagen a grises para que la librería de opencv trabaje de
# una forma más óptima
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Se cambia la representación de la imagen a una en forma binaria utilizando
# THRESH_BINARY para cambiar los valores de umbralización a lo
# máximo que es 255, eliminando las iluminaciones variables.
_, threshold = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)

#findContours es una función que permite buscar los n contornos de una imagen bin
```

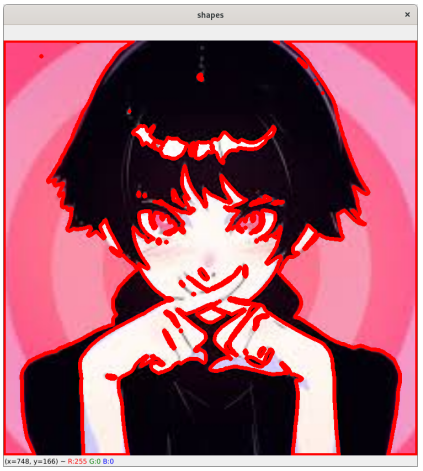
```
contours, _ = cv2.findContours(
    threshold, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

#Crea un contador del tipo int
i = 0

#Inicia un ciclo
for contour in contours:  }
    #Si i es igual a 0; i se convierte en 1
    if i == 0:
        i = 1
        continue

     #Es la función que según los parámetros que tenga puede buscar los números de vértice de una forma para saber qué forma geométrica es
    approx = cv2.approxPolyDP(
        #Se le da los parámetros para la forma geométrica
        contour, 0.01 * cv2.arcLength(contour, True), True)

    #Se dibuja el contorno de la forma con el color rojo en RGB
    cv2.drawContours(img, [contour], 0,(0, 0, 255), 5)

      #Calcula todos los momentos hasta el tercer orden de un polígono o forma rasterizada
    M = cv2.moments(contour)

     #Si el valor de la posición m00 es diferente de 0.0 se le asigna a x y y nuevos valores de tipo entero
    if M['m00'] != 0.0:
        #Asignación de x
        x = int(M['m10']/M['m00'])
        #Asignación de y
        y = int(M['m01']/M['m00'])
```

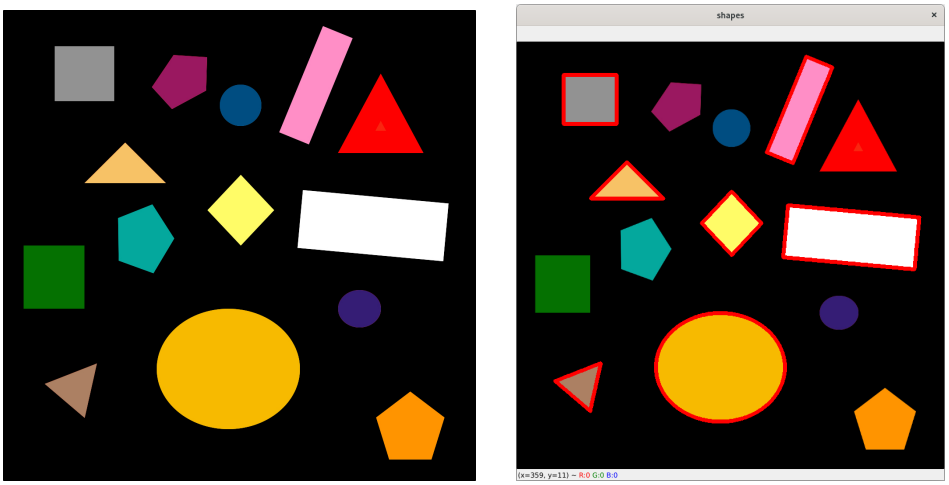
```
#Muestra en un frame la imagen con los dibujos en sus contornos
cv2.imshow('shapes', img)
#Se queda en espera a una entrada
cv2.waitKey(0)
#Destruye el frame creado
cv2.destroyAllWindows()
```

Pruebas de ejecución

Se utilizaron 4 imágenes obtenidas de google para verificar el funcionamiento de la librería opencv en conjunto de sus métodos, los resultados fueron los siguientes:







En las dichas imágenes de funcionamiento se aprecia el funcionamiento de la librería opencv al detectar formas geométricas, pero su funcionamiento puede variar con cada imagen.

Como se aprecia puede detectar formas donde es imperceptible el ver una forma, así mismo no detectar una forma debido a su diseño (Anchura y angostura).

Conclusión

La implementación de estos trabajos fue sencilla y nueva para nosotros, ya que se tenía un conocimiento previo de lo que se estaba haciendo, puesto que anteriormente se había utilizado Python.

Lo interesante que se aprendió ahora es la manera en la que se puede programar una red neuronal para poder obtener los resultados deseados es decir en el datos de los billetes podemos obtener una respuesta satisfactoria con un porcentaje de clasificación de los valores correctos, y también pudimos apreciar cómo se pueden reconocer figuras dentro de una imagen de manera eficiente.

Referencias

[1] Colaboradores de los proyectos Wikimedia. "Teoría de la resonancia adaptativa - Wikipedia, la enciclopedia libre". Wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Teoría_de_la_resonancia_adaptativa (accedido el 9 de octubre de 2022).

[2] "Redes Neuronales en java, Herramienta para redes neuronales. Diseño de redes neuronales en Java: JRedesNeuronales - tutorial de redes neuronales herramienta java Marco de trabajo. http://www.redes-neuronales.com.es/tutorial-redes-neuronales/tutorial-redes.htm (accedido el 10 de octubre de 2022).

[3] "How to Create a Multilayer Perceptron Neural Network in Python - Technical Articles". All About Circuits - Electrical Engineering & Electronics Community. https://www.allaboutcircuits.com/technical-articles/how-to-create-a-multilayer-perceptron-neural-network-in-python/ (accedido el 12 de octubre de 2022).

[4] "Python OpenCV | cv2.imread() method - GeeksforGeeks". GeeksforGeeks. https://www.geeksforgeeks.org/python-opencv-cv2-imread-method/#:~:text=OpenCV

-Python%20is%20a%20library%20of%20Python%20bindings%20designed,format)%20then%20this%20method%20returns%20an%20empty%20matrix. (accedido el 10 de octubre de 2022).

[5] "La librería Pandas | Aprende con Alf". Aprende con Alf. https://aprendeconalf.es/docencia/python/manual/pandas/ (accedido el 14 de octubre de 2022).

[6] M. Zippo. "Método Python OpenCV cv2.line ()". The best Python programming books to read in 2021. Learn Python at Python.Engineering. https://python.engineering/es_sp-python-opencv-cv2-line-method/ (accedido el 7 de octubre de 2022).

[7] "Uso de Cv2 en python - programador clic". programador clic. https://programmerclick.com/article/7125211453/ (accedido el 6 de octubre de 2022).

[8] "Cómo Detectar Figuras Geométricas en OpenCV - DataSmarts". DataSmarts. https://www.datasmarts.net/como-detectar-figuras-geometricas-en-opencv/ (accedido el 10 de octubre de 2022).