Contents

1 Sistema Inteligente para detección de digitos escritas mediante gestos

El sistema inteligente propuesto es capaz de reconocer dígitos obtenidos mediante métodos de visión por computador. Es decir, útilizando el dedo índice de una mano, y "dibujando en el aire", el sistema será capaz de reconocer el dígito dibujado. La aplicación se ejecuta de la siguiente manera:

1. Se abrirán dos ventanas
   (a) Ventana 1 (**"Camera Capture"**): La primera Ventana contendrá el video en tiempo real capturado por la webcam mediante las OpenCV
   (b) Ventana 2 (**"MNIST Image"**): Esta ventana contendrá la imagen que posteriormente será procesada para reconocer el dígito que se quiere evaluar.
2. El usuario tiene 3 opciones:
   (a) Pulsar **"d"**: Alternará entre modo Dibujar y modo No Dibujar.
   (b) Pulsar **"m"**: Pasará la imagen mostrada en la ventana "MNIST Image" por varios filtros para procesarla con el modelo de IA y reconocer el dígito dibujado
   (c) Pulsar **"q"**: Salir de la aplicación

1.1 Herramientas necesarias

Para poder llevar a cabo la implementación de la idea se hará uso de las siguientes herramientas:

- *python*
- *opencv*
- *mediapipe*
- *numpy*
- *tensorflow*
- *keras*

1.1.1 Resolución de dependencias

Para resolver las dependencias de las mismas es suficiente con instalar ***python*** y ***pip*** en tu sistema y ejecutar los siguientes comandos:

```
pip install opencv-python
pip install opencv-contrib-python
pip install mediapipe
pip install numpy
pip install tensorflow
pip install keras
```

La idea principal es mediante la libreria opencv capturar video desde una cámara web. Para posteriormente filtrar las imagenes y pasarlas por el modelo de mediapipe para detecciónn de manos, lo que permitirá conocer la posición de cada dedo. De esta manera será más fácil administrar el reconocimiento de gestos. Además de utilizar un método de ML, también se hará uso de las técnicas de segmentación y descripción vistas en clase así como de preproceso y filtrado.

2 Para entrenar el modelo y generar model.h5

```
cd mnist_model
python mnist.py
```

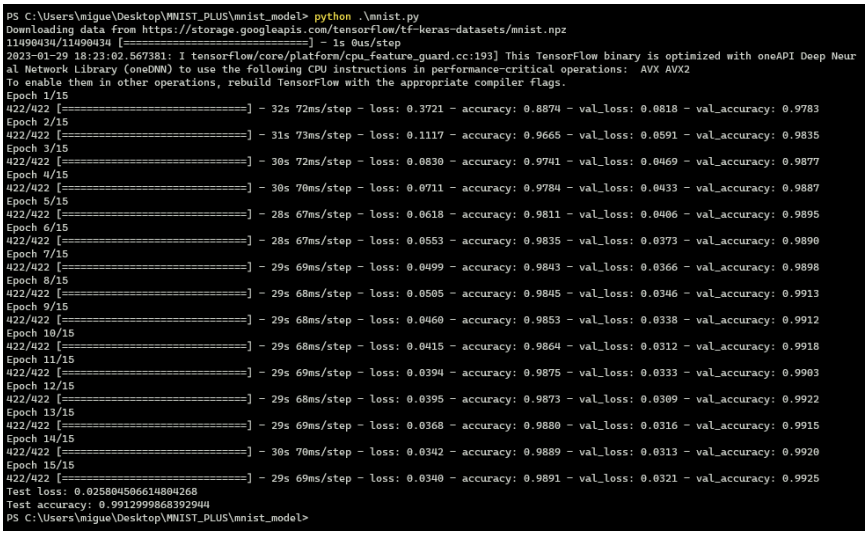

Figure 1: salida de consola después de entrenar el modelo

3 Para probar el modelo utilizando el ratón como medio para dibujar

```
python testmodel.py
```

4 Para utilizar el modelo ya entrenado y probado con mediapipe y el reconocimiento de manos

```
python main.py
```

5 Aspectos concretos del proyecto

El proyecto consta de 3 ficheros con extensión *".py"*

```
- mnist.py # Corresponde con el código para generar el modelo "model.h5". Es decir, es
- testmodel # Se trata de un entorno de pruebas que he utilizado para comprobar que el
- main.py #El fichero que contiene el programa completo incluye las funcionalidades ap
- model.h5 # Modelo generado por mnist.py.
```

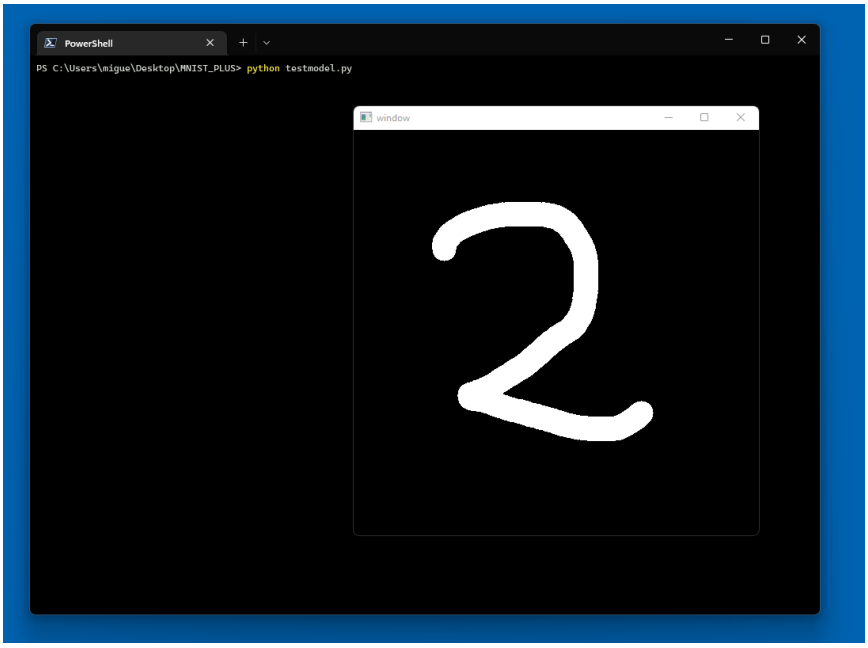

Figure 2: Ventana para dibujar con el ratón

```
PS C:\Users\migue\Desktop\MNIST_PLUS> python testmodel.py
2023-01-29 18:58:39.032710: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neur
al Network Library (oneDNN) to use the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
1/1 [==============================] - 0s 240ms/step
prediction =  [2]
PS C:\Users\migue\Desktop\MNIST_PLUS>
```

Figure 3: Predicción realizada sobre la imagen después de reescalarla a 28x28 (Pulsar ESC una vez para mostrar la imagen 28x28 y volver a pulsar para realizar la predicción)

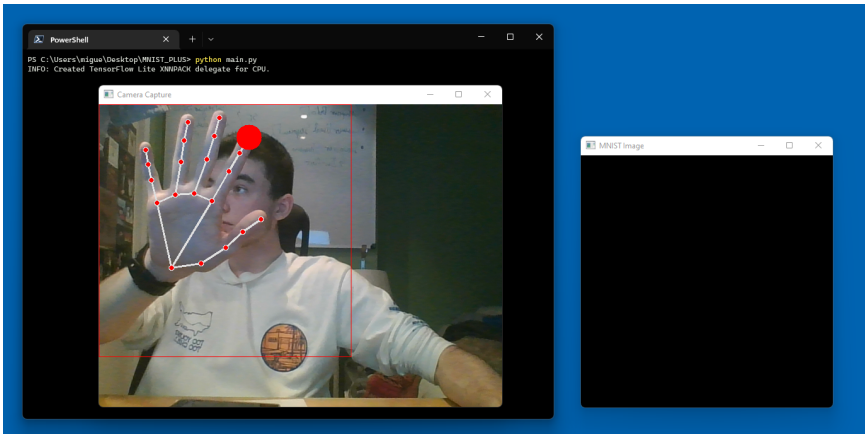

Figure 4: Aparecen las ventanas para dibujar

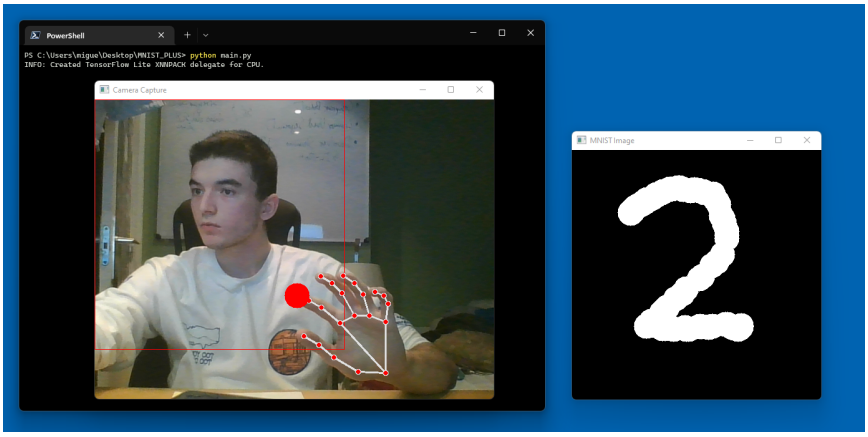

Figure 5: Pulsando "D" pasamos a modo Dibujar, realizamos el dibujo y volviendo a pulsar "D" salimos del modo dibujo

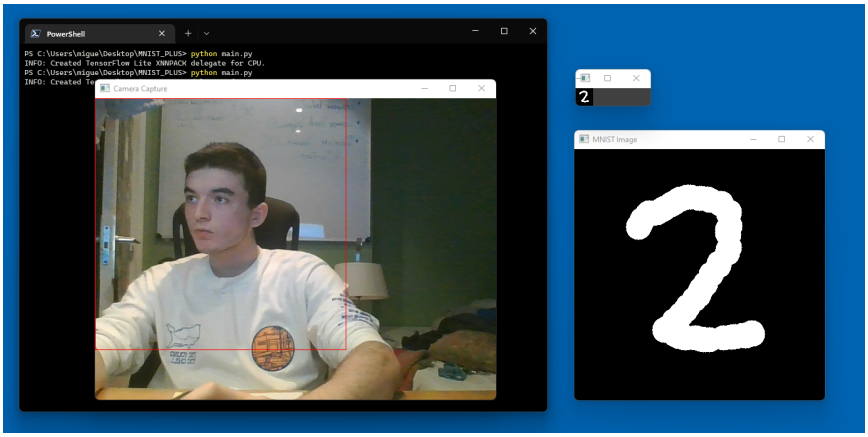

Figure 6: Pulsando "M" aplicamos el modelo a la imagen reescalada a 28x28

```
PS C:\Users\migue\Desktop\MNIST_PLUS> python main.py
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
2023-01-29 19:13:12.271489: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimize
d with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical op
erations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
1/1 [==============================] - 0s 280ms/step
prediccion =  [[4.3034873e-12 5.1068527e-09 9.9991536e-01 8.4405838e-05 7.6143835e-17
  2.5185476e-15 4.8628251e-13 1.9645980e-09 2.7367435e-07 1.2226539e-12]]
prediction =  [2]
PS C:\Users\migue\Desktop\MNIST_PLUS>
```

Figure 7: Predicción del sistema. Pulsar "M" otra vez

Todos los fichero fuente listados contienen comentarios suficientemente descriptivos como para entender el funcionamiento de los diversos elementos que componen la aplicación