TFM_LauraRivera_objetivo2_similares

June 18, 2023

<img src="http://www.uoc.edu/portal/_resources/common/imatges/marca_UOC/UOC_Masterbrand.jpg", align="left">

Identificación de huellas de calzado a partir de imágenes con redes neuronales convolucionales

MU Ingeniería Computacional y Matemática / Área de Inteligencia Artificial

Laura Rivera Sanchez

1 Objetivo 2: Búsqueda de huellas similares para la identificación de individuos

El objetivo de este cuaderno es crear una red neuronal siamesa para obtener imágenes similares a la de entrada.

Para ello se utiliza el conjunto de datos disponible en: https://fid.dmi.unibas.ch/

Dispone de dos tipos de imágenes: las de referencia (muestras tomadas a personas) y las "tracks" (recogidas en escenas de crimen), de éstas últimas se dispone también de su versión recortada (cropped), que es la que se utilizará también en los experimentos.

[2]:
```
#librerias necesarias:
import pandas as pd
import numpy as np
from zipfile import ZipFile
import matplotlib.pyplot as plt
from PIL import Image
import os
import cv2
import random
import skimage
```

[3]:
```
#En caso de utilizar google colab:
#from google.colab import drive
#drive.mount('/content/gdrive')
```

1.1 Lectura y análisis de los conjuntos de datos FD-300

[4]:
```
def get_images_to_jpeg(imgPath):
  dir_list = os.listdir("./"+imgPath)
  result = []

  for f in dir_list:

    im = Image.open("./"+imgPath+"/"+f)
    im2=im.resize((400,912))
    im3 = im2.crop((40,40,320,872)) #Quitar marco medidor
    im3.save("./"+imgPath+"/"+f[0:-4]+'jpeg')

    result.append(f[0:-4]+'jpeg')
    os.remove("./"+imgPath+"/"+f)

  print('Nº files:',len(result))
  return result

def get_images(imgPath):
  dir_list = os.listdir("./"+imgPath)
  result = []
  for f in dir_list:
    result.append(f)


  print('Nº files:',len(result))
  return result
```

[5]:
```
with ZipFile('data/FID300/images.zip', 'r') as zipObj:
  zipObj.extractall('fid300')
```

[6]:
```
#Lectura de la tabla de resultados
df_fid300 = pd.read_csv('data/FID300/label_table.csv', delimiter=',')
```

En este caso, las imágenes estan divididas en 3 carpetas:

- *tracks_original*: Conteine las 300 imágenes originales extraidas de la escena del crimen, incluyendo la plantilla de medición.
- *tracks_cropped*: Contiene las imágenes originales pero ya recortadas.
- *references*: Contiene las 1175 imágenes de las huellas de referencia, obtenidas de diferentes calzados.

Para el proyecto se decide utilizar las dos últimas categorias, guardandolas en 2 datasets diferenciados: *fid300ref* y *fid300crop*.

1.1.1 Análisis de los datos

[7]:
```
fid300ref = get_images("fid300/references")
fid300crop = get_images("fid300/tracks_cropped")

print(len(df_fid300)) #lineas en el csv
print(fid300ref[0])
```

```
Nº files: 1175
Nº files: 300
300
00481.png
```

[8]:
```
X_files = df_fid300['X'].values.tolist()
y = df_fid300['y'].values.tolist()

values_y, counts_y = np.unique(y, return_counts=True)

print('Nº of references: %d' %len(values_y))

dfref = pd.DataFrame({'x':values_y, 'y':counts_y})
dfref = dfref.sort_values('y', ascending = False)
dfref.head(10) # coger los 5 primeros
```

```
Nº of references: 130
```

[8]:

	x	y
32	37	7
19	22	6
2	3	6
4	5	6
44	49	6
12	14	6
89	1055	6
79	1041	5
55	60	5
9	11	5

1.1.2 Visualización de imágenes

Se ha creado la función *plot_image* que permite la visualización de las imágenes de cualquiera de las dos bases de datos.

Parámetros:
imgPath: carpeta donde estan las imágenes
fileNames: array con los nombres de los ficheros a mostrar

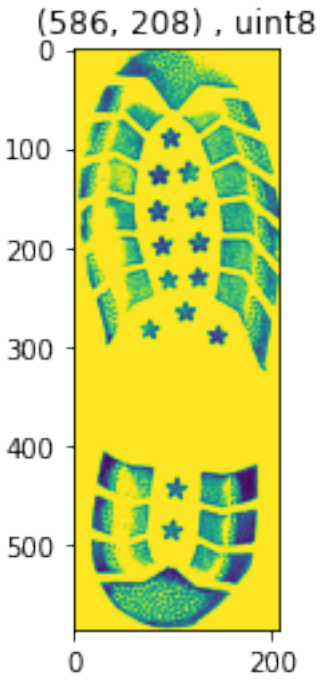
[9]:
```
import skimage
def plot_image(imgPath, fileNames):
  for i in range(len(fileNames)):
```

```
    filename = fileNames[i]
    img = skimage.io.imread(imgPath+filename)

    plt.figure()
    plt.title(str(img.shape)+" , "+str(img.dtype))
    plt.imshow(img)
  print(fileNames)
  plt.show()

def plot_image2(img):

    plt.figure()
    plt.title(str(img.shape)+" , "+str(img.dtype))
    plt.imshow(img)

    plt.show()
def plot_image_grey(img):

    plt.figure()
    plt.title(str(img.shape)+" , "+str(img.dtype))
    plt.imshow(img, cmap='gray')

    plt.show()
```

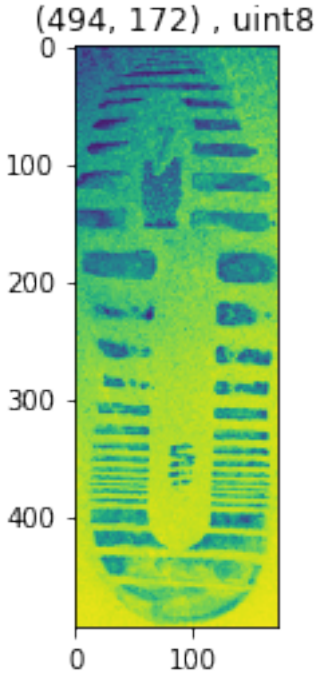
A continuación, se muestra una imagen aleatória de cada uno de los 3 datasets del proyecto:

[10]:
```
plot_image("fid300/references/",random.choices(fid300ref,k=1))
plot_image("fid300/tracks_cropped/",random.choices(fid300crop,k=1))
```

```
['00478.png']
```


(586, 208) , uint8
0
100
200
300
400
500
0
200

['00168.jpg']


(494, 172) , uint8
0
100
200
300
400
0
100

1.1.3 División de los datos

En este caso, no es necesaria la división de datos, ya que se va a utilizar todas las imágenes de referencia para codificar sus características en la red neuronal, representando las imágenes que podría la polícia tener en su base de datos (de personas conocidas).

1.2 Red neuronal siamesa

Red neuronal siamesa para la búsqueda de imágenes similares.

[13]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator, load_img,␣
↪img_to_array, array_to_img
from tensorflow.keras.models import Model, load_model
from tensorflow.keras.layers import Flatten, Conv2D, Conv2DTranspose,␣
↪LeakyReLU, BatchNormalization, Input, Dense, Reshape, Activation
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ModelCheckpoint
import tensorflow.keras.backend as K
import numpy as np
import matplotlib.pyplot as plt
from tqdm import tqdm_notebook as tqdm
import pickle
import pandas as pd
import tensorflow as tf

# Load images
img_height = 586
img_width = 156
channels = 1 #3
batch_size = 16

def my_preprocessing_function(img):

    #Combines all the transformations
    #img = cv.imread('fid300/'+filename)
    #img = rgb2gray(img)
    test_binary_high,img = cv.threshold(img,0, 255, cv2.THRESH_BINARY)
    #img = crop_image(img)
    #img = cv2.resize(img, (img_height, img_width), interpolation = cv2.
↪INTER_AREA)
    img = np.expand_dims(img,-1)
   # img = gray2rgb(img)
    return img



train_datagen = ImageDataGenerator(preprocessing_function␣
↪=my_preprocessing_function, rescale=1./255,
```

```
                                    validation_split=0.2)

#añadir funcion preprocessing_function para pasar a gris y binary
training_set = train_datagen.flow_from_directory(
    'fid300/',
     target_size = (img_height, img_width),
     batch_size = batch_size,
     classes=['references'],
     class_mode = 'input',
     subset = 'training',
     color_mode='grayscale',
     shuffle=True)

validation_set = train_datagen.flow_from_directory(
    'fid300/',
     target_size = (img_height, img_width),
     batch_size = batch_size,
     classes=['references'],
     subset = 'validation',
     class_mode = 'input',
     color_mode='grayscale',
     shuffle=False)

#Única clase porque no tengo classes / labels por imagen.
```

```
Found 940 images belonging to 1 classes.
Found 235 images belonging to 1 classes.
```

[18]:
```
# Define the autoencoder
input_model = Input(shape=(img_height, img_width, channels))

print(input_model.shape)
# Encoder layers
encoder = Conv2D(32, (3,3), padding='same',␣
 ↪kernel_initializer='normal')(input_model)
encoder = LeakyReLU()(encoder)
encoder = BatchNormalization(axis=-1)(encoder)

encoder = Conv2D(64, (3,3), padding='same',␣
 ↪kernel_initializer='normal')(encoder)
encoder = LeakyReLU()(encoder)
encoder = BatchNormalization(axis=-1)(encoder)

encoder = Conv2D(64, (3,3), padding='same',␣
 ↪kernel_initializer='normal')(input_model)
encoder = LeakyReLU()(encoder)
encoder = BatchNormalization(axis=-1)(encoder)
```

```
encoder_dim = K.int_shape(encoder)
encoder = Flatten()(encoder)

# Latent Space
latent_space = Dense(16, name='latent_space')(encoder)

# Decoder Layers
decoder = Dense(np.prod(encoder_dim[1:]))(latent_space)
decoder = Reshape((encoder_dim[1], encoder_dim[2], encoder_dim[3]))(decoder)

decoder = Conv2DTranspose(64, (3,3), padding='same',␣
 ↪kernel_initializer='normal')(decoder)
decoder = LeakyReLU()(decoder)
decoder = BatchNormalization(axis=-1)(decoder)

decoder = Conv2DTranspose(64, (3,3), padding='same',␣
 ↪kernel_initializer='normal')(decoder)
decoder = LeakyReLU()(decoder)
decoder = BatchNormalization(axis=-1)(decoder)

decoder = Conv2DTranspose(32, (3,3), padding='same',␣
 ↪kernel_initializer='normal')(decoder)
decoder = LeakyReLU()(decoder)
decoder = BatchNormalization(axis=-1)(decoder)

decoder = Conv2DTranspose(1, (3, 3), padding="same")(decoder)
output = Activation('sigmoid', name='decoder')(decoder)

print(output.shape)
```

```
(None, 586, 156, 1)
(None, 586, 156, 1)
```

[19]:
```
# Create model object
autoencoder = Model(input_model, output, name='autoencoder')
autoencoder.summary()
```

```
Model: "autoencoder"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 586, 156, 1)]     0

 conv2d_5 (Conv2D)           (None, 586, 156, 64)      640

 leaky_re_lu_8 (LeakyReLU)   (None, 586, 156, 64)      0
```

```
batch_normalization_8 (Batc  (None, 586, 156, 64)     256
hNormalization)

flatten_1 (Flatten)          (None, 5850624)          0

latent_space (Dense)         (None, 16)               93610000

dense_1 (Dense)              (None, 5850624)          99460608

reshape_1 (Reshape)          (None, 586, 156, 64)     0

conv2d_transpose_4 (Conv2DT  (None, 586, 156, 64)     36928
ranspose)

leaky_re_lu_9 (LeakyReLU)    (None, 586, 156, 64)     0

batch_normalization_9 (Batc  (None, 586, 156, 64)     256
hNormalization)

conv2d_transpose_5 (Conv2DT  (None, 586, 156, 64)     36928
ranspose)

leaky_re_lu_10 (LeakyReLU)   (None, 586, 156, 64)     0

batch_normalization_10 (Bat  (None, 586, 156, 64)     256
chNormalization)

conv2d_transpose_6 (Conv2DT  (None, 586, 156, 32)     18464
ranspose)

leaky_re_lu_11 (LeakyReLU)   (None, 586, 156, 32)     0

batch_normalization_11 (Bat  (None, 586, 156, 32)     128
chNormalization)

conv2d_transpose_7 (Conv2DT  (None, 586, 156, 1)      289
ranspose)

decoder (Activation)         (None, 586, 156, 1)      0

=================================================================
Total params: 193,164,753
Trainable params: 193,164,305
Non-trainable params: 448
_________________________________________________________________
```

[20]:
```
!pip install opencv-contrib-python==4.4.0.44
```

```
Requirement already satisfied: opencv-contrib-python==4.4.0.44 in
/Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/site-packages
(4.4.0.44)
Requirement already satisfied: numpy>=1.17.3 in
/Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/site-packages
(from opencv-contrib-python==4.4.0.44) (1.23.4)

[notice] A new release of pip is
available: 23.0.1 -> 23.1.2
[notice] To update, run:
pip install --upgrade pip
```

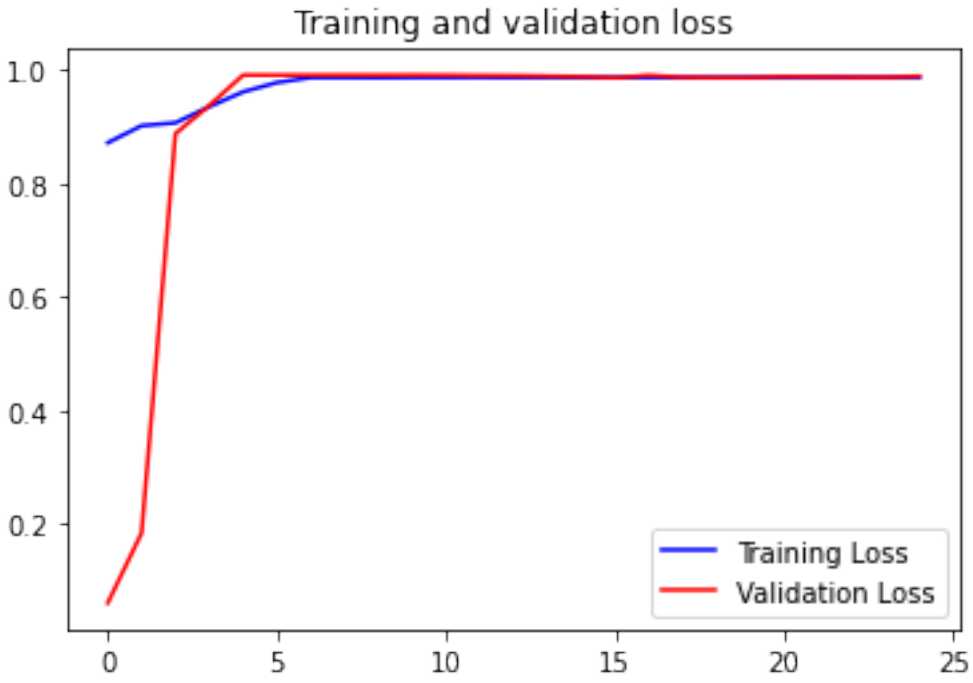
[22]:
```
# Compile the model
autoencoder.compile(loss="mse", optimizer= Adam(learning_rate=1e-3),␣
 ↪metrics=['accuracy'])
#autoencoder.compile(optimizer='adam',
#               loss=tf.keras.losses.
 ↪SparseCategoricalCrossentropy(from_logits=True),
#               metrics=['accuracy'])
# Fit the model
history = autoencoder.fit(
          training_set,
          steps_per_epoch=training_set.n // batch_size,
          epochs=25,
          validation_data=validation_set,
          validation_steps=validation_set.n // batch_size,
          callbacks = [ModelCheckpoint('models/image_autoencoder_2.h5',
                                       monitor='val_loss',
                                       verbose=0,
                                       save_best_only=True,
                                       save_weights_only=False)])
```

```
Epoch 1/25
58/58 [==============================] - 643s 11s/step - loss: 0.1348 -
accuracy: 0.8717 - val_loss: 0.9292 - val_accuracy: 0.0597
Epoch 2/25
58/58 [==============================] - 601s 10s/step - loss: 0.0996 -
accuracy: 0.9018 - val_loss: 0.7845 - val_accuracy: 0.1832
Epoch 3/25
58/58 [==============================] - 603s 10s/step - loss: 0.0924 -
accuracy: 0.9069 - val_loss: 0.1121 - val_accuracy: 0.8877
Epoch 4/25
58/58 [==============================] - 603s 10s/step - loss: 0.0646 -
accuracy: 0.9353 - val_loss: 0.0610 - val_accuracy: 0.9374
Epoch 5/25
58/58 [==============================] - 601s 10s/step - loss: 0.0386 -
accuracy: 0.9614 - val_loss: 0.0103 - val_accuracy: 0.9911
Epoch 6/25
```

```
58/58 [==============================] - 604s 10s/step - loss: 0.0223 -
accuracy: 0.9779 - val_loss: 0.0097 - val_accuracy: 0.9911
Epoch 7/25
58/58 [==============================] - 609s 11s/step - loss: 0.0129 -
accuracy: 0.9868 - val_loss: 0.0091 - val_accuracy: 0.9911
Epoch 8/25
58/58 [==============================] - 603s 10s/step - loss: 0.0126 -
accuracy: 0.9868 - val_loss: 0.0089 - val_accuracy: 0.9911
Epoch 9/25
58/58 [==============================] - 601s 10s/step - loss: 0.0126 -
accuracy: 0.9867 - val_loss: 0.0088 - val_accuracy: 0.9909
Epoch 10/25
58/58 [==============================] - 594s 10s/step - loss: 0.0124 -
accuracy: 0.9870 - val_loss: 0.0086 - val_accuracy: 0.9910
Epoch 11/25
58/58 [==============================] - 600s 10s/step - loss: 0.0125 -
accuracy: 0.9869 - val_loss: 0.0087 - val_accuracy: 0.9909
Epoch 12/25
58/58 [==============================] - 575s 10s/step - loss: 0.0123 -
accuracy: 0.9871 - val_loss: 0.0088 - val_accuracy: 0.9903
Epoch 13/25
58/58 [==============================] - 592s 10s/step - loss: 0.0125 -
accuracy: 0.9868 - val_loss: 0.0091 - val_accuracy: 0.9900
Epoch 14/25
58/58 [==============================] - 581s 10s/step - loss: 0.0124 -
accuracy: 0.9870 - val_loss: 0.0093 - val_accuracy: 0.9894
Epoch 15/25
58/58 [==============================] - 577s 10s/step - loss: 0.0124 -
accuracy: 0.9869 - val_loss: 0.0097 - val_accuracy: 0.9884
Epoch 16/25
58/58 [==============================] - 572s 10s/step - loss: 0.0124 -
accuracy: 0.9870 - val_loss: 0.0113 - val_accuracy: 0.9863
Epoch 17/25
58/58 [==============================] - 575s 10s/step - loss: 0.0124 -
accuracy: 0.9869 - val_loss: 0.0088 - val_accuracy: 0.9908
Epoch 18/25
58/58 [==============================] - 581s 10s/step - loss: 0.0124 -
accuracy: 0.9869 - val_loss: 0.0109 - val_accuracy: 0.9870
Epoch 19/25
58/58 [==============================] - 576s 10s/step - loss: 0.0124 -
accuracy: 0.9869 - val_loss: 0.0111 - val_accuracy: 0.9867
Epoch 20/25
58/58 [==============================] - 580s 10s/step - loss: 0.0122 -
accuracy: 0.9871 - val_loss: 0.0111 - val_accuracy: 0.9866
Epoch 21/25
58/58 [==============================] - 577s 10s/step - loss: 0.0121 -
accuracy: 0.9872 - val_loss: 0.0098 - val_accuracy: 0.9886
Epoch 22/25
```

```
58/58 [==============================] - 571s 10s/step - loss: 0.0120 -
accuracy: 0.9874 - val_loss: 0.0103 - val_accuracy: 0.9880
Epoch 23/25
58/58 [==============================] - 574s 10s/step - loss: 0.0120 -
accuracy: 0.9873 - val_loss: 0.0108 - val_accuracy: 0.9872
Epoch 24/25
58/58 [==============================] - 571s 10s/step - loss: 0.0121 -
accuracy: 0.9872 - val_loss: 0.0109 - val_accuracy: 0.9871
Epoch 25/25
58/58 [==============================] - 575s 10s/step - loss: 0.0122 -
accuracy: 0.9871 - val_loss: 0.0095 - val_accuracy: 0.9892
```

[23]:
```python
# Plot Accuracy and Loss
loss = history.history['accuracy']
val_loss = history.history['val_accuracy']

epochs = range(len(loss))

plt.figure()

plt.plot(epochs, loss, 'b', label='Training Acc')
plt.plot(epochs, val_loss, 'r', label='Validation Acc')
plt.title('Training and validation accuracy')
plt.legend()

plt.show()
```



[14]:
```
# Predict image function
def predict_image(image_dir, autoencoder):
  raw_image = load_img(image_dir, target_size=(img_height, img_width),␣
 ↪color_mode = "grayscale")
  image = img_to_array(raw_image)
  image = np.expand_dims(image, axis=0)
  image = image / 255.0
  pred = autoencoder.predict(image)
  pred = pred * 255.0
  pred = np.reshape(pred, (img_height, img_width, 1))
  pred = array_to_img(pred)

  return raw_image, pred
```

[ ]:
```
# Predict 3 images from training set
train_imgs = training_set.filepaths[0:3]

for i in train_imgs:
  raw_image, pred_image = predict_image(i, autoencoder)

  # Show original image
  plt.imshow(raw_image)
  plt.show()

  # Show predicted image
  plt.imshow(pred_image)
  plt.show()
```

[ ]:
```
autoencoder.save('models/image_autoencoder_2.h5')
```

[16]:
```
import os
import cv2 as cv
#Latent space:

autoencoder = load_model('models/image_autoencoder_2.h5', compile=False)
latent_space_model = Model(autoencoder.input, autoencoder.
 ↪get_layer('latent_space').output)

#
# Load all images and predict them with the latent space model
X = []
indices = []

for i in tqdm(range(len(os.listdir('./fid300/references')))):
  try:
```

```
    img_name = os.listdir('./fid300/references')[i]
    img = load_img('./fid300/references/{}'.format(img_name),
                   target_size = (img_height, img_width), color_mode =␣
↪"grayscale")
    img = img_to_array(img) / 255.0
    test_binary_high,img = cv.threshold(img,0, 255, cv2.THRESH_BINARY)
    #img = crop_image(img)
    #img = cv2.resize(img, (img_height, img_width), interpolation = cv2.
↪INTER_AREA)
    #img = np.expand_dims(img,-1)

    #img = cv2.resize(img, (img_height, img_width), interpolation = cv2.
↪INTER_AREA)
    img = np.expand_dims(img, axis=0)
    pred = latent_space_model.predict(img)
    pred = np.resize(pred, (16))
    X.append(pred)
    indices.append(img_name)

  except Exception as e:
    print(img_name)
    print(e)

    # Export the embeddings
embeddings = {'indices': indices, 'features': np.array(X)}
pickle.dump(embeddings,
            open('./fid300/image_embeddings.pickle', 'wb'))
#
def eucledian_distance(x,y):
  eucl_dist = np.linalg.norm(x - y)
  return eucl_dist
```

```
<ipython-input-16-2083651edf6d>:13: TqdmDeprecationWarning: This function will
be removed in tqdm==5.0.0
Please use `tqdm.notebook.tqdm` instead of `tqdm.tqdm_notebook`
  for i in tqdm(range(len(os.listdir('./fid300/references')))):
```

```
HBox(children=(FloatProgress(value=0.0, max=1175.0), HTML(value='')))
```

```
1/1 [==============================] - 0s 276ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 67ms/step
```

```
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 72ms/step
1/1 [==============================] - 0s 74ms/step
1/1 [==============================] - 0s 75ms/step
1/1 [==============================] - 0s 71ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
```

```
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 71ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
```

```
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
```

```
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
```

```
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 70ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 71ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 63ms/step
```

```
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 70ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 70ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
```

```
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
```

```
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
```

```
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 59ms/step
```

```
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 76ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
```

```
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 70ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
```

```
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 92ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 66ms/step
```

```
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 70ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 70ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
```

```
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 64ms/step
```

```
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
```

```
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 56ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 71ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 72ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 82ms/step
```

```
1/1 [==============================] - 0s 82ms/step
1/1 [==============================] - 0s 72ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 70ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 72ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 71ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
```

```
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
```

```
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 75ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
```

```
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 76ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 61ms/step
```

```
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 75ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 62ms/step
```

```
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
```

```
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
```

```
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 73ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 71ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 72ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 63ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 64ms/step
```

```
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
```

[17]:
```python
# Load embeddings
embeddings = pickle.load(open('./fid300/image_embeddings.pickle', 'rb'))

# Load images metadata
metadata = pd.read_csv('data/FID300/label_table.csv')
```

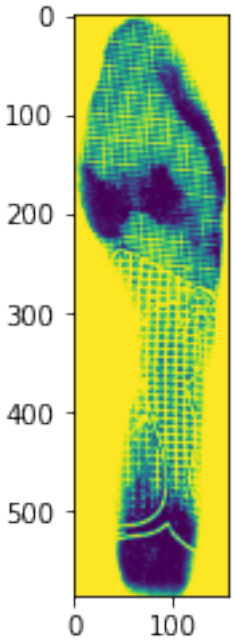
[60]:
```python
def getTop3Similars(img):
    # Calculate eucledian distance
    img_similarity = []
    #cat_similarity = []

    # Get actual image embedding
    img = img_to_array(img) / 255.0
    test_binary_high,img = cv2.threshold(img,0, 255, cv2.THRESH_BINARY)
    img = np.expand_dims(img, axis=0)

    pred = latent_space_model.predict(img)
    pred = np.resize(pred, (16))

    for i in range(len(embeddings['indices'])):
      img_name = embeddings['indices'][i]

      # Calculate vectors distances
      dist = eucledian_distance(pred,embeddings['features'][i])
      img_similarity.append(dist)


    imgs_result = pd.DataFrame({'img': embeddings['indices'],
                                'euclidean_distance': img_similarity})
```

```
    imgs_result = imgs_result.query('euclidean_distance > 0').
↪sort_values(by='euclidean_distance', ascending=True).reset_index(drop=True)

    print("Most similar images")
    fig = plt.figure(figsize=(10, 7))
    fig.tight_layout(pad=50.0)

    # Show 3 most similar images
    for i in range(3):
      image = load_img('./fid300/references/{}'.format(imgs_result['img'].
↪values[i]))

      # Show image
      fig.add_subplot(1, 3, i+1)
      plt.imshow(image)
      plt.title('{}'.format(imgs_result['img'].values[i])+"\n"+'Dist: {}'.
↪format(imgs_result['euclidean_distance'].values[i]))


    plt.show()
    print("Less similar images")
    fig = plt.figure(figsize=(10, 7))
    fig.tight_layout(pad=50.0)

    # Show 3 less similar images
    less = imgs_result.tail(3)
    for i in range(3):
        image = load_img('./fid300/references/{}'.format(less['img'].values[i]))

        # Show image
        fig.add_subplot(1, 3, i+1)
        plt.imshow(image)
        plt.title('{}'.format(less['img'].values[i])+"\n"+'Dist: {}'.
↪format(less['euclidean_distance'].values[i]))
    plt.show()
```

1.2.1 Experimentos con las imágenes de referencia

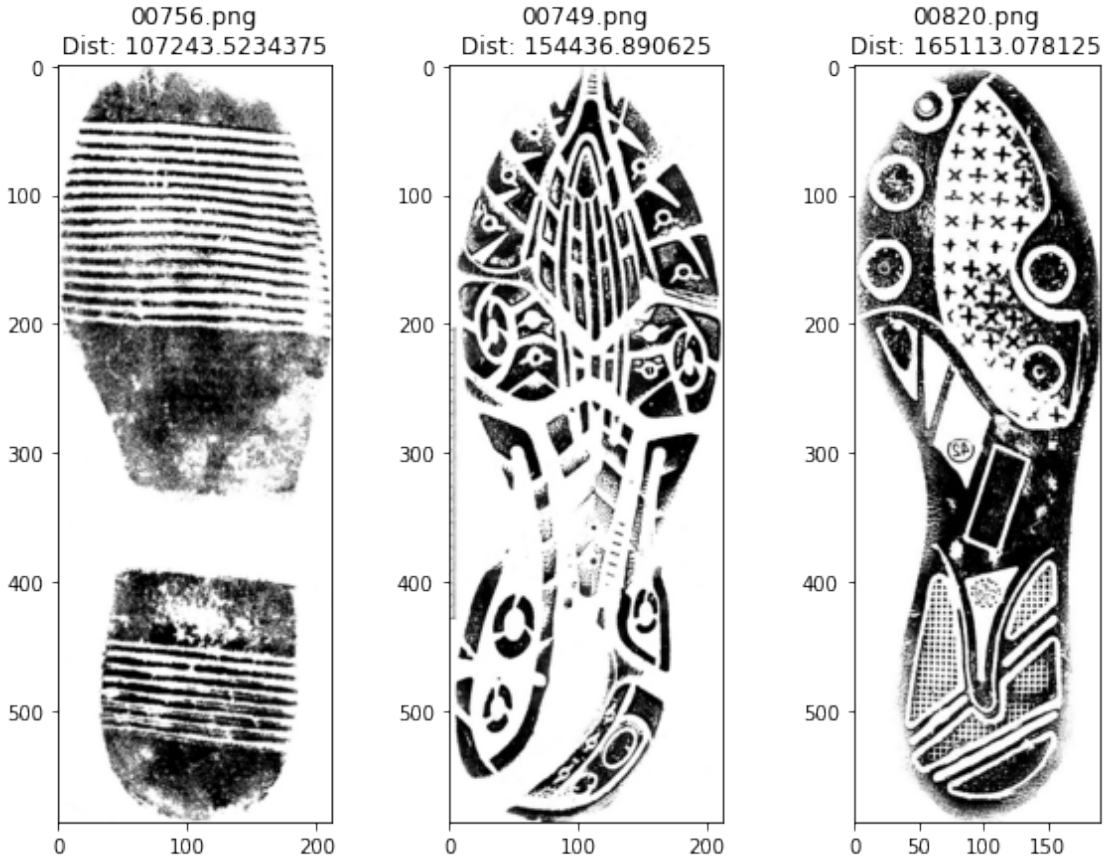
[61]:
```
# Get random image name
img_name = os.listdir('./fid300/references')[random.randint(0, 299)]
test1 = load_img('./fid300/references/{}'.format(img_name),
                 target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

#my_preprocessing_function(img)

plt.imshow(test1)
```

[61]:
```
<matplotlib.image.AxesImage at 0x7fca62b934f0>
```

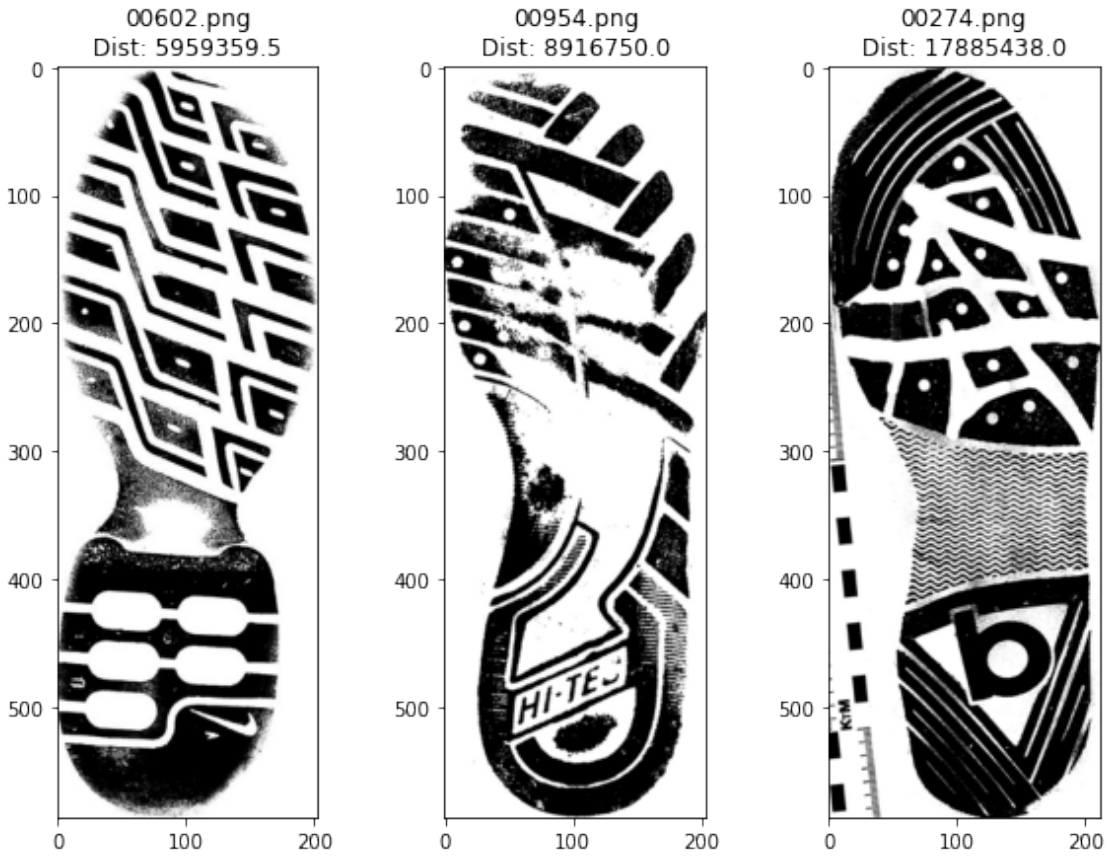


[62]:
```
getTop3Similars(test1)
```

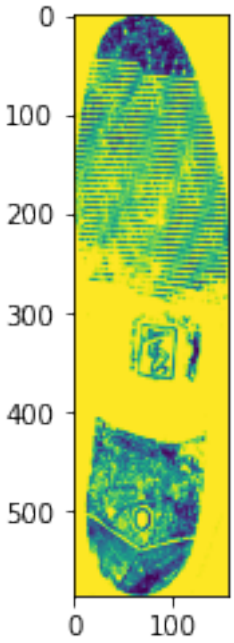
```
1/1 [==============================] - 0s 75ms/step
Most similar images
```



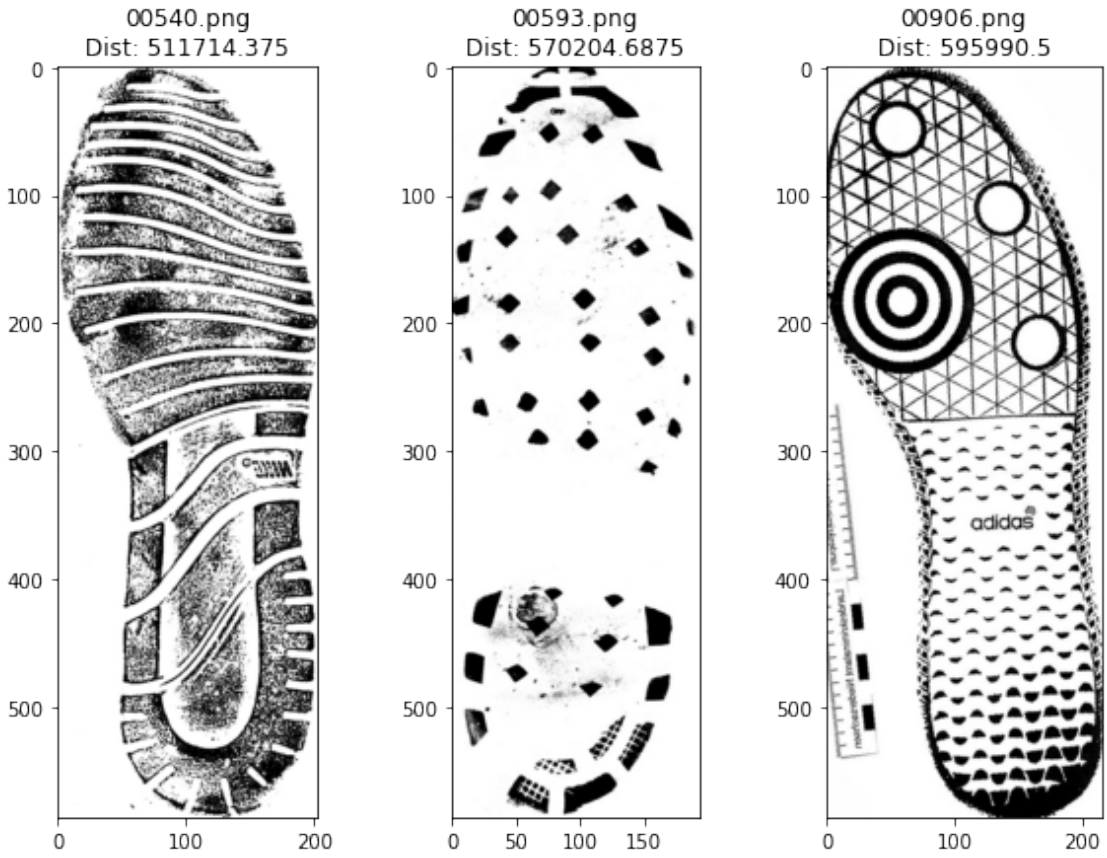
Less similar images



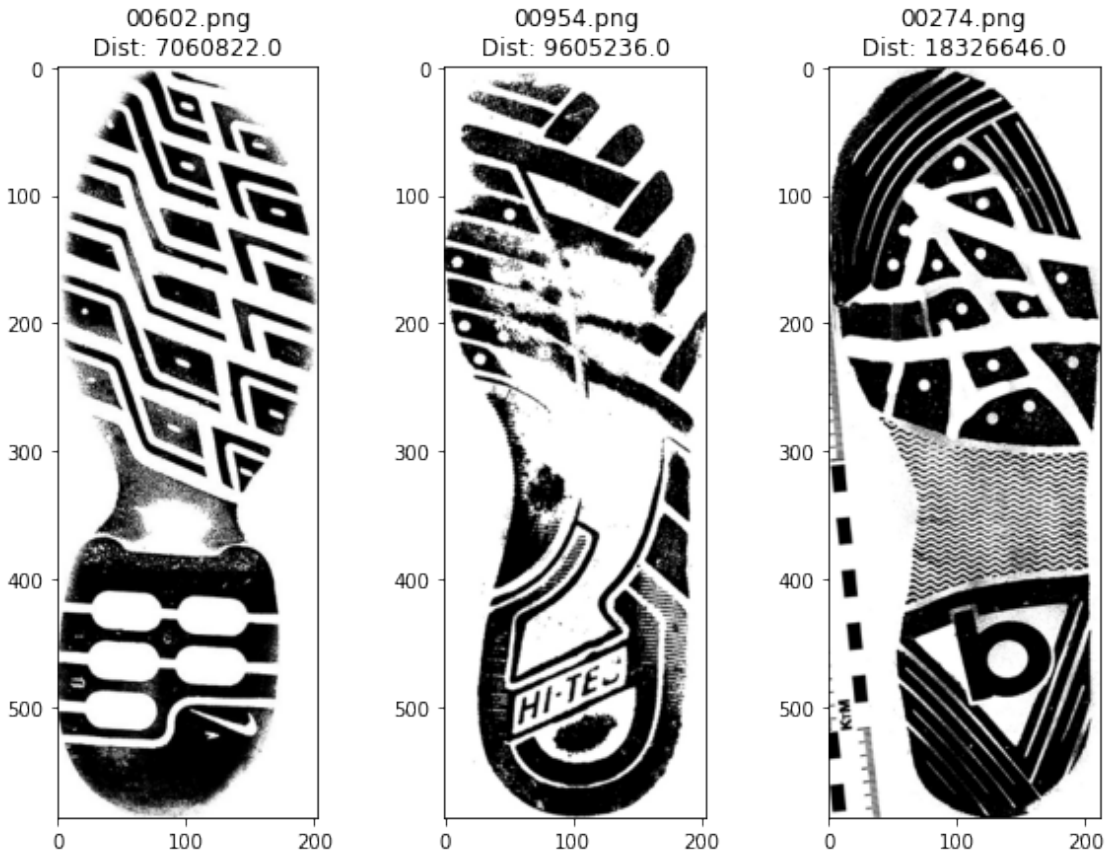
[63]:
```
#Test 00755.png
test00755= load_img('./fid300/references/{}'.format("00755.png"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test00755)
plt.show()
getTop3Similars(test00755)
```



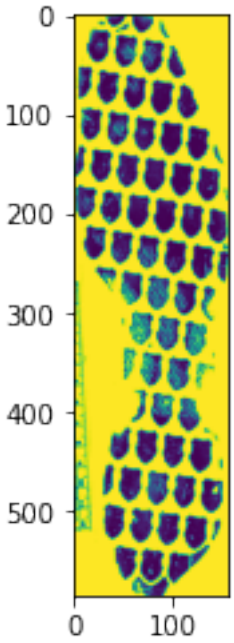

```
1/1 [==============================] - 0s 80ms/step
Most similar images
```



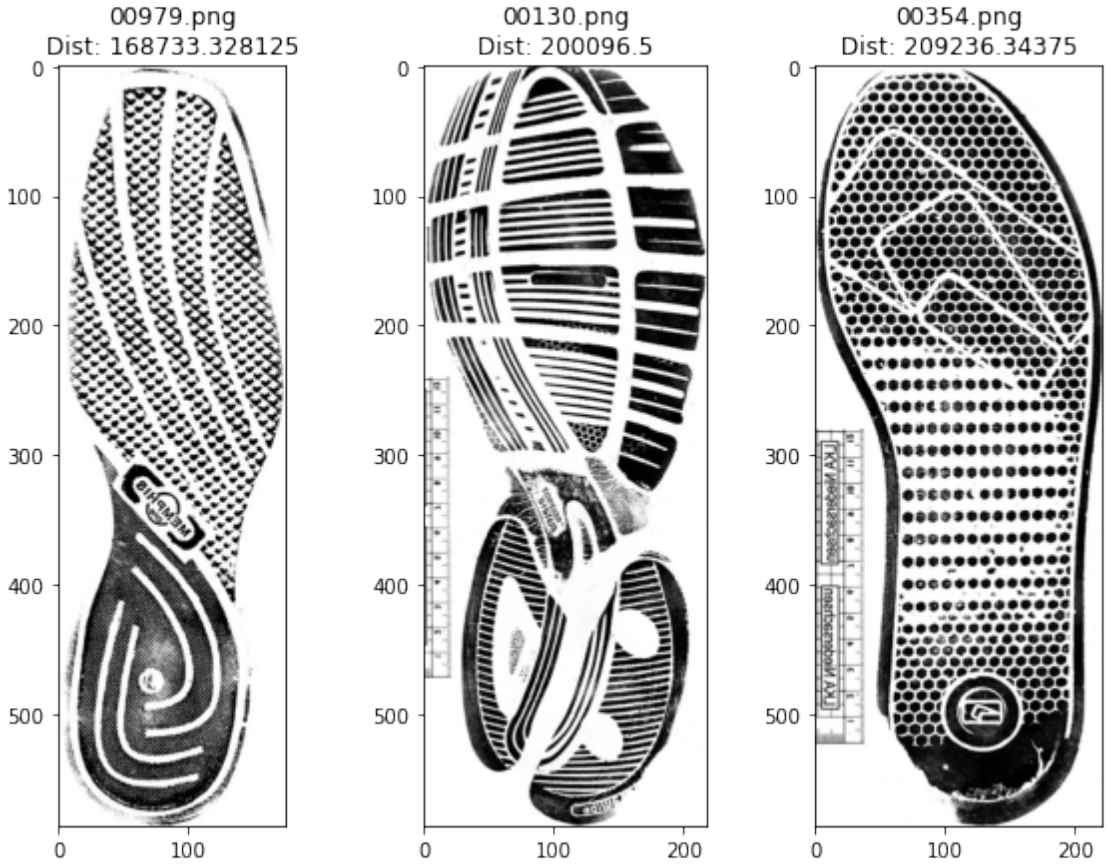
Less similar images



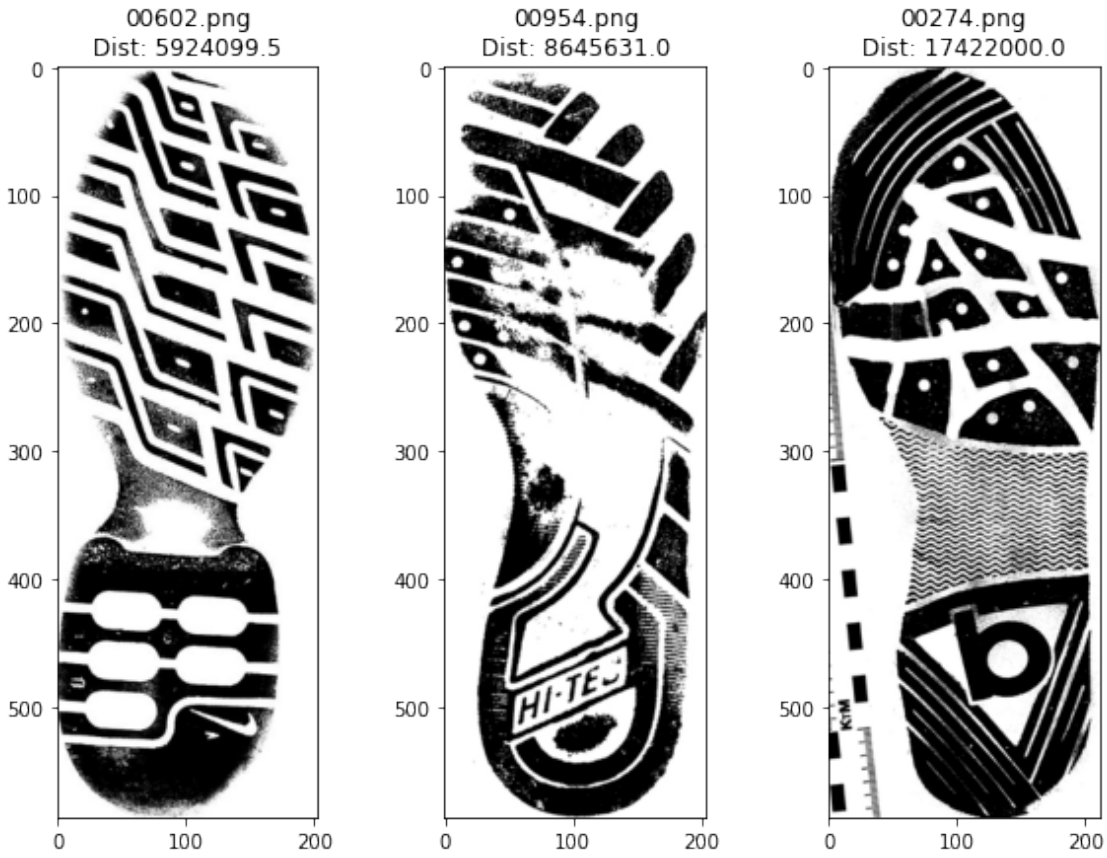
[64]:
```
#Test 00403.png

test00403= load_img('./fid300/references/{}'.format("00403.png"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test00403)
plt.show()
getTop3Similars(test00403)
```



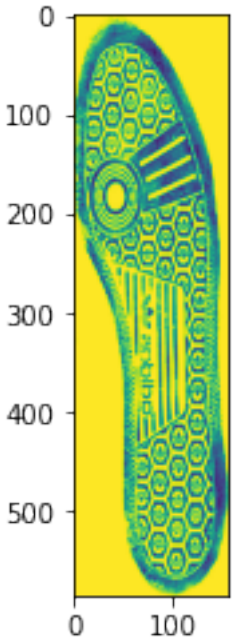

```
1/1 [==============================] - 0s 75ms/step
Most similar images
```



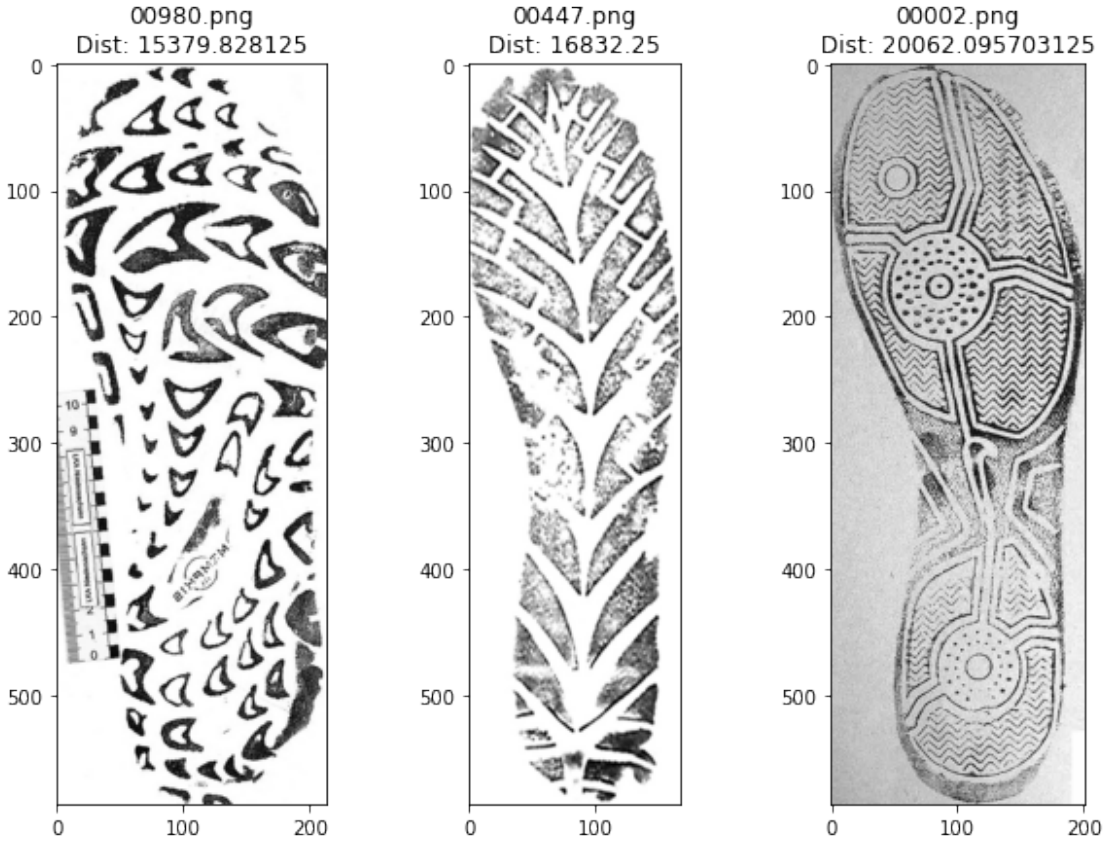
Less similar images



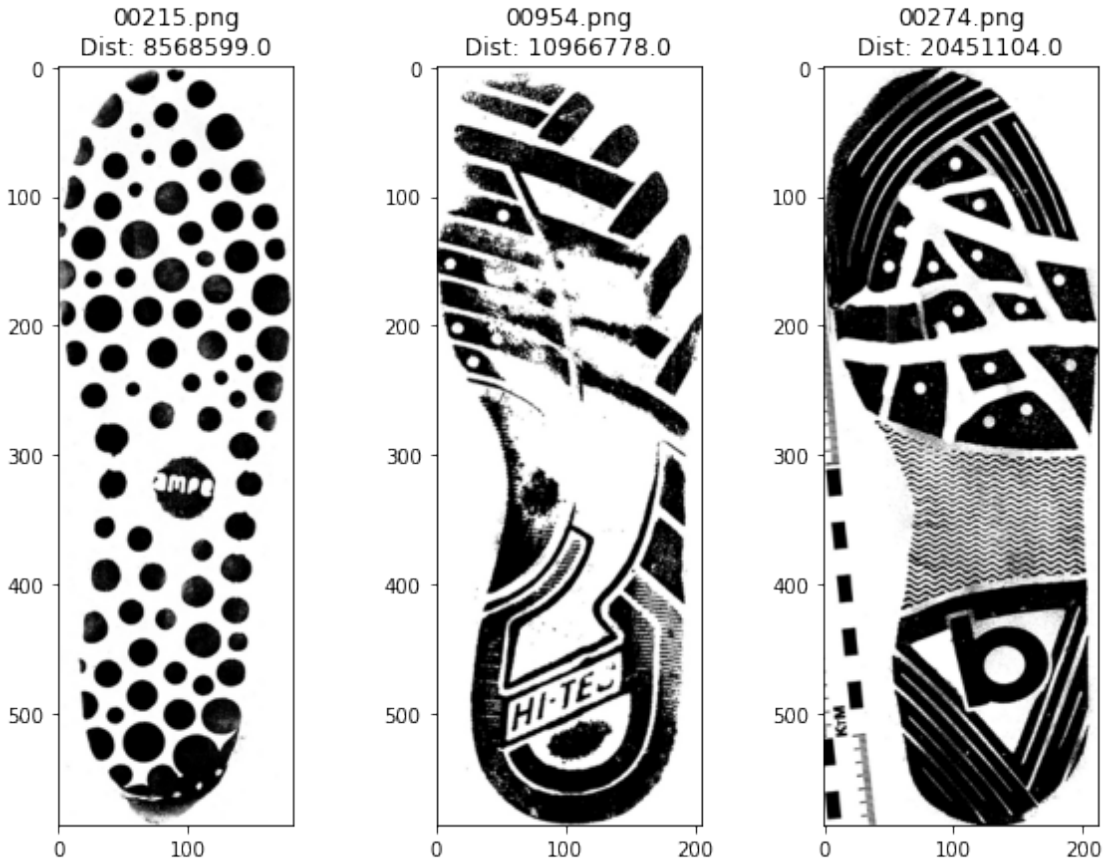
[65]:
```
#Test 00109.png

test00109= load_img('./fid300/references/{}'.format("00109.png"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test00109)
plt.show()
getTop3Similars(test00109)
```



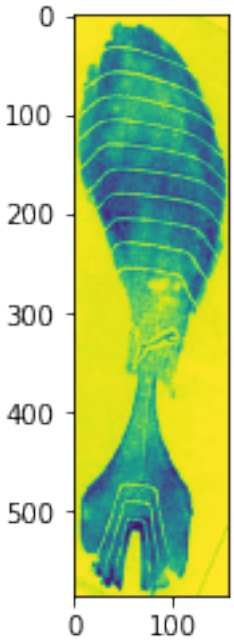

```
1/1 [==============================] - 0s 73ms/step
Most similar images
```



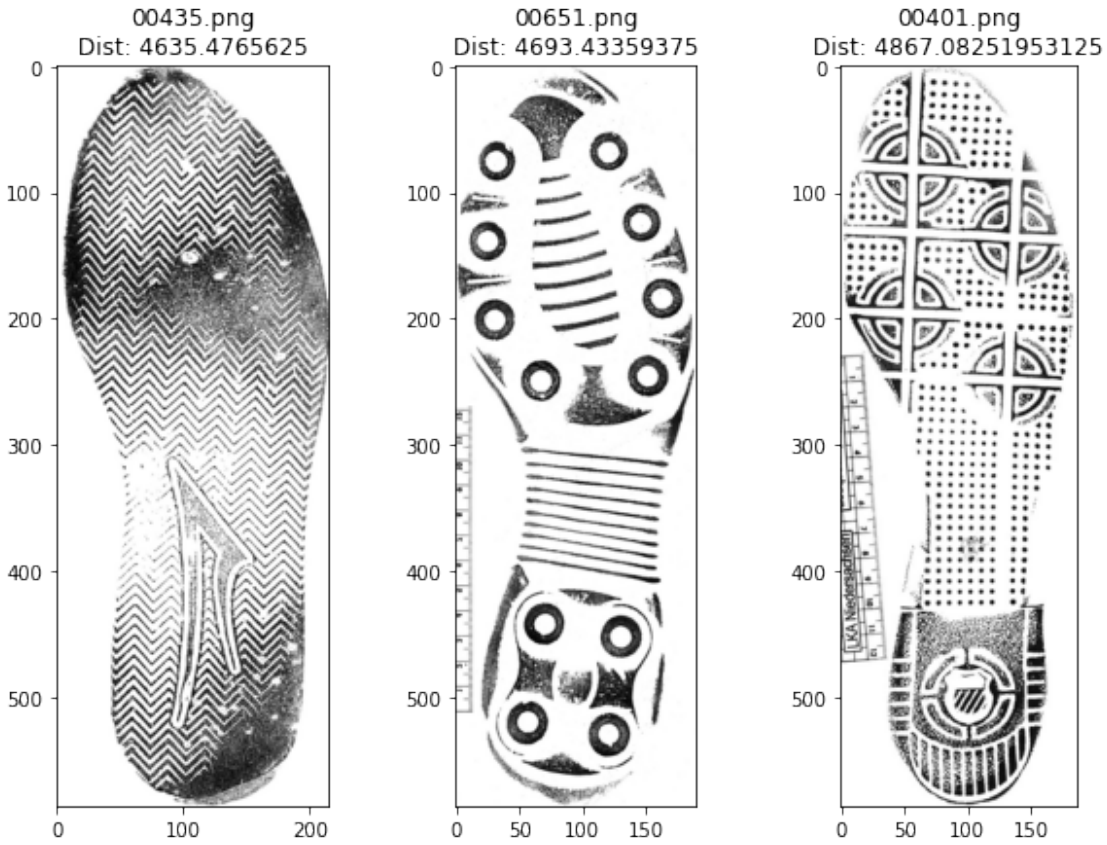
Less similar images



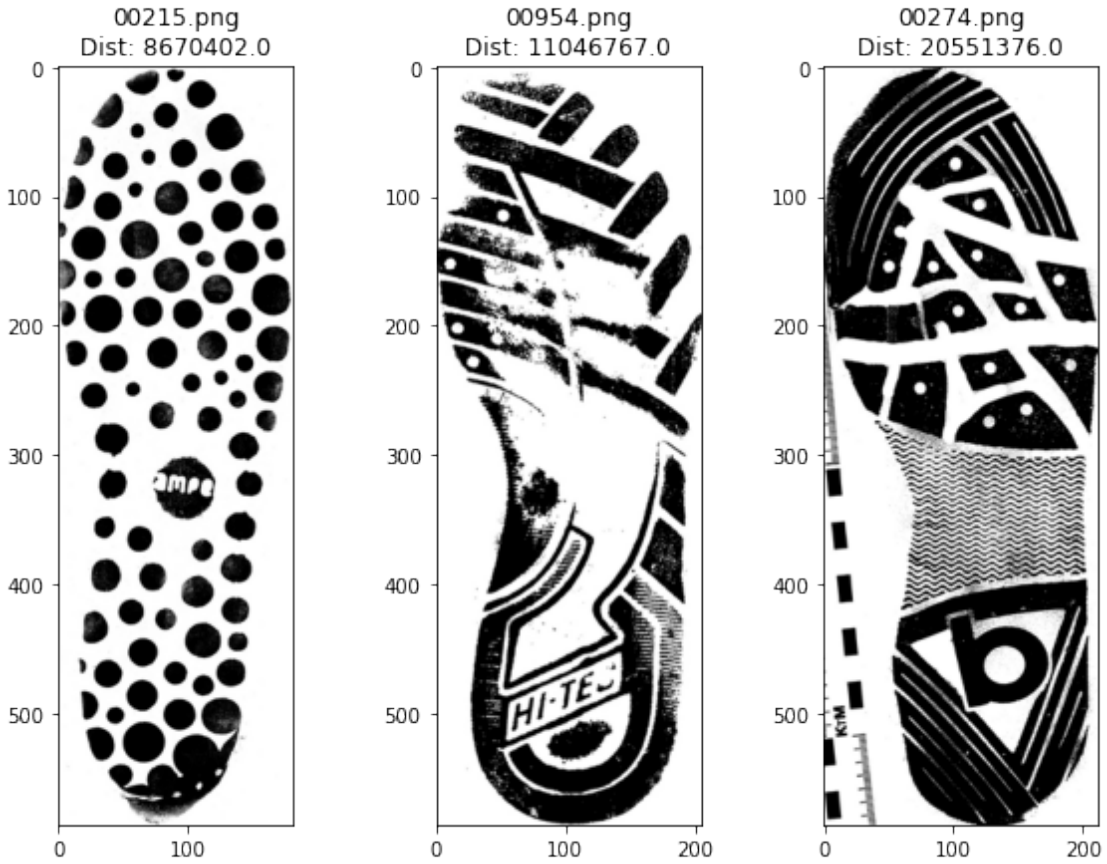
[66]:
```
#Test 00033.png
test00033= load_img('./fid300/references/{}'.format("00033.png"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test00033)
plt.show()
getTop3Similars(test00033)
```



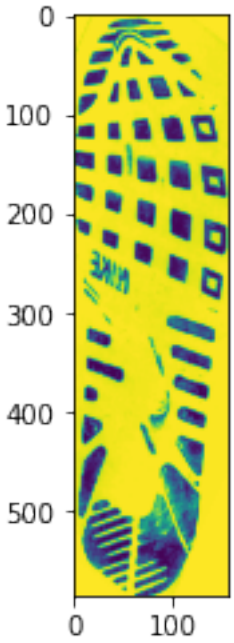

```
1/1 [==============================] - 0s 69ms/step
Most similar images
```



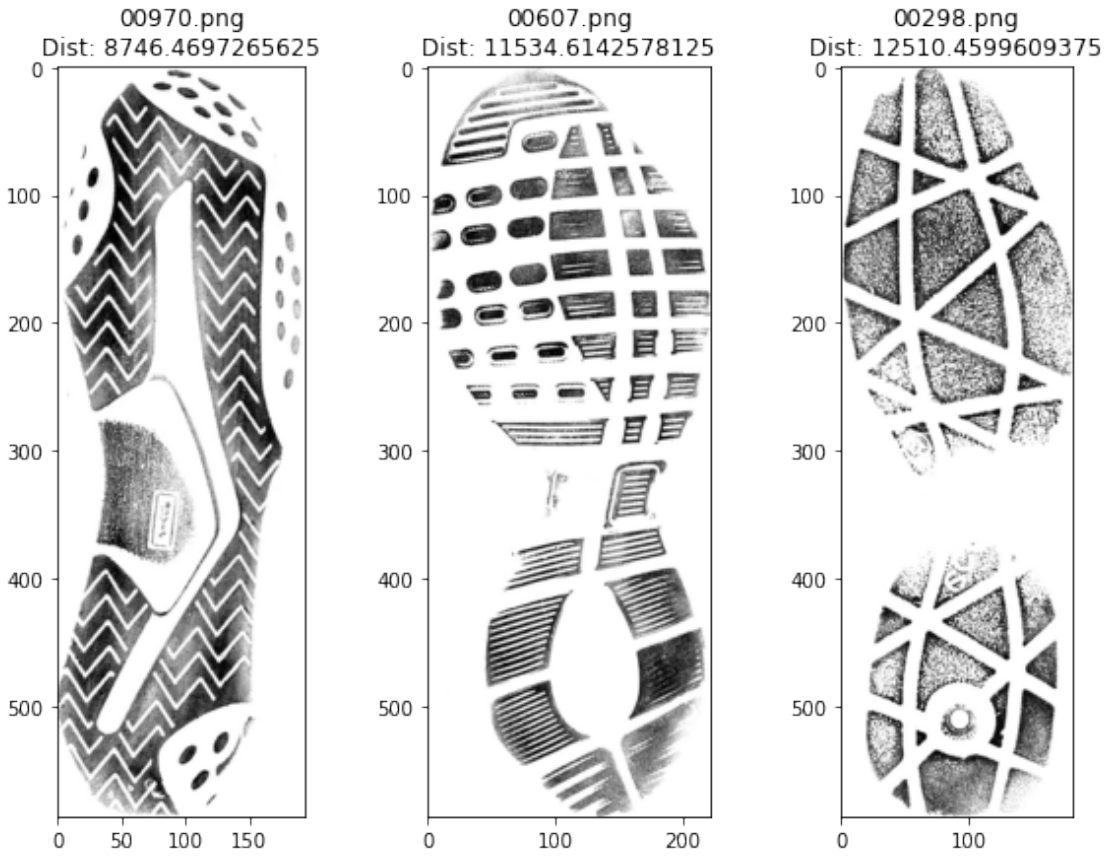
Less similar images



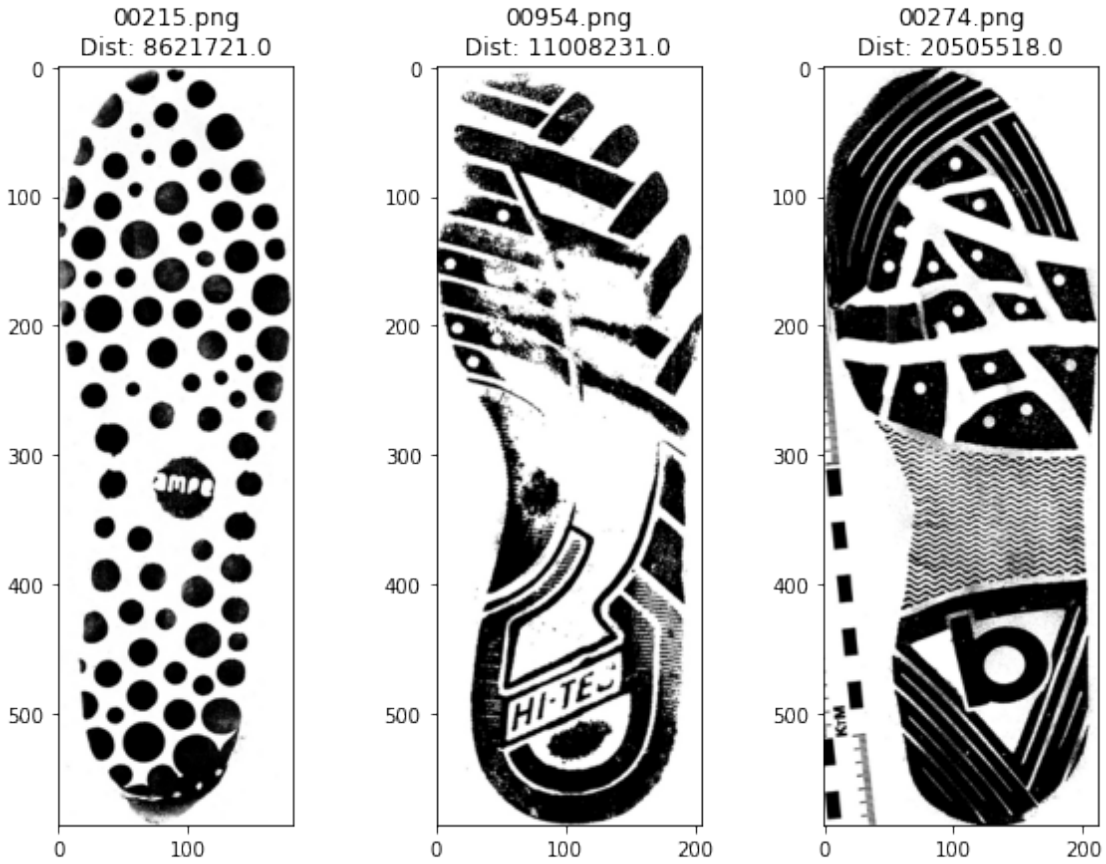
[67]:
```
#Test 00011.png
test00011= load_img('./fid300/references/{}'.format("00011.png"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test00011)
plt.show()
getTop3Similars(test00011)
```




```
1/1 [==============================] - 0s 68ms/step
Most similar images
```



Less similar images

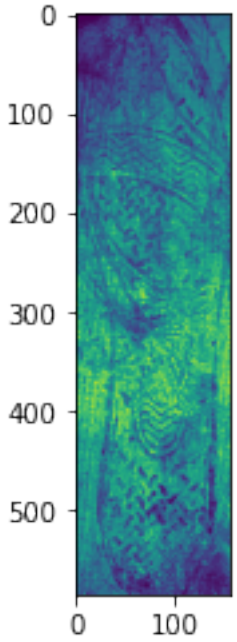


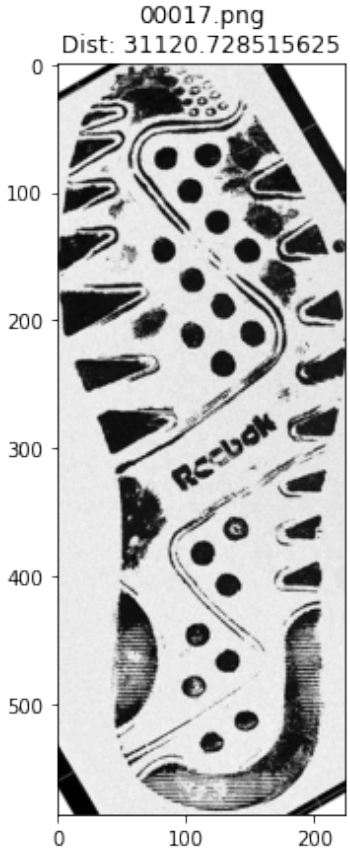
1.2.2 Experimentos con imágenes de huellas reales

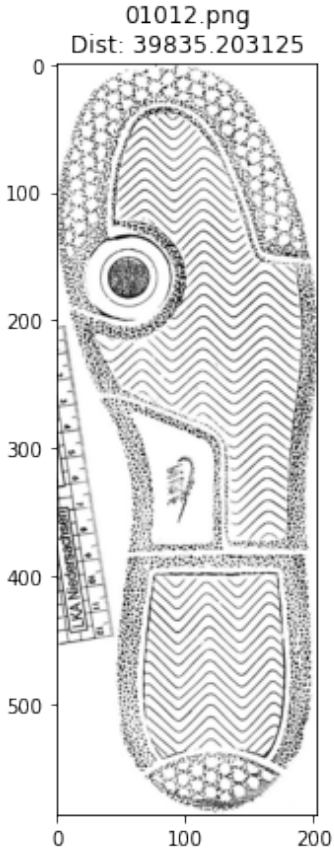
[73]:
```
import re
# Get image name
img_name = os.listdir('./fid300/tracks_cropped')[random.randint(0, 299)]
test2 = load_img('./fid300/tracks_cropped/{}'.format(img_name),
                 target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test2)
code = re.sub("^0+(?!$)","",img_name[0:5])

solution = df_fid300[df_fid300.X == int(code)]
print ("Testing image: "+ code + " Suggested solution: "+str(solution['y'].
 ↪iloc[0]))

getTop3Similars(test2)
```

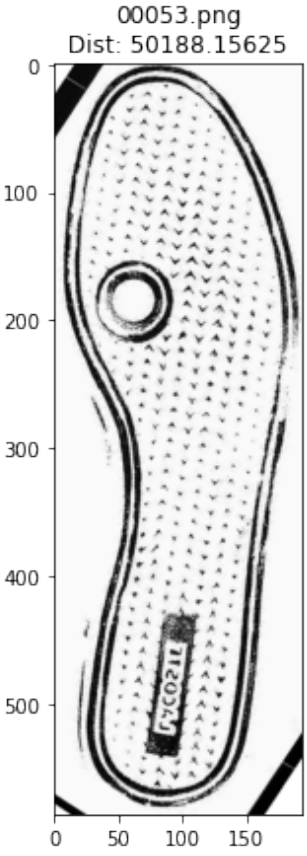
```
Testing image: 67 Suggested solution: 45
1/1 [==============================] - 0s 72ms/step
```

Most similar images

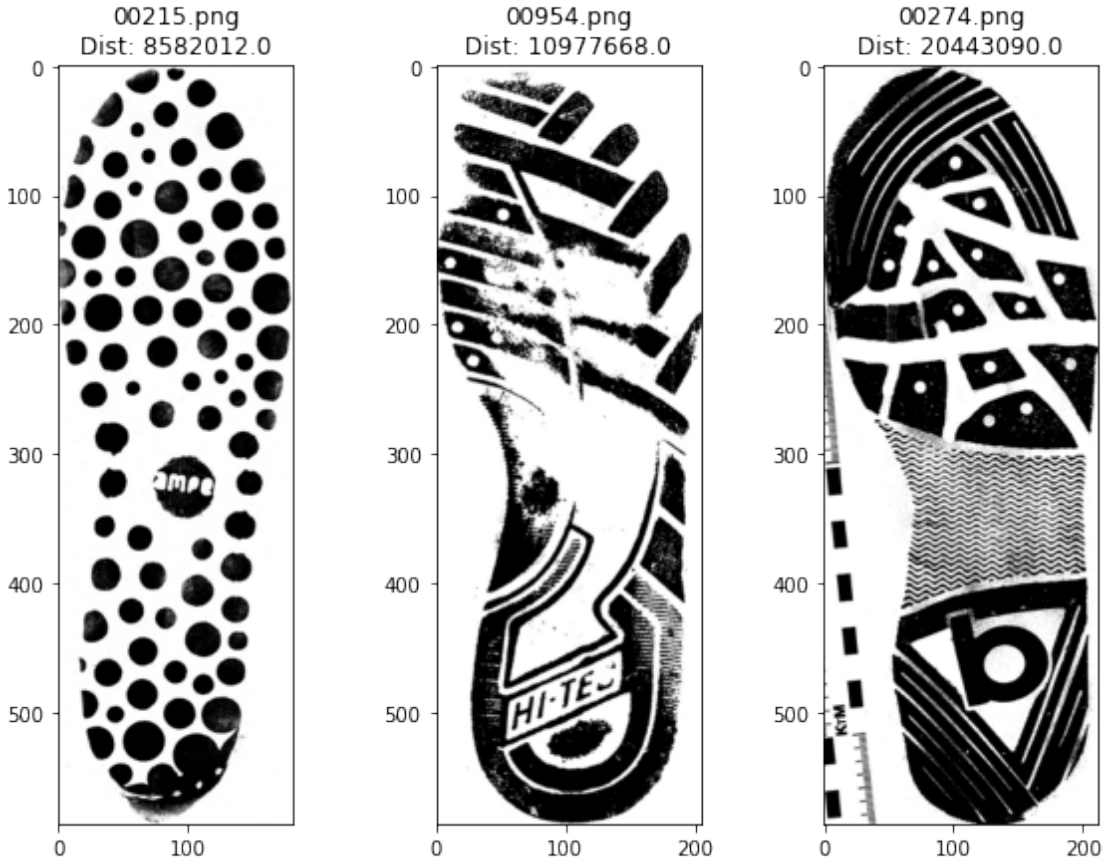




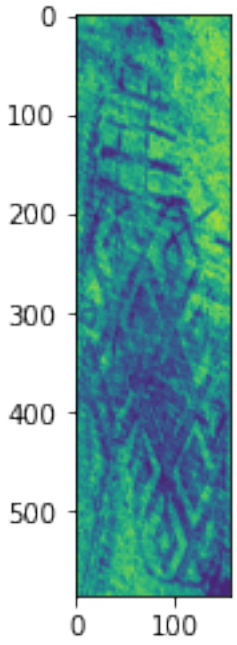




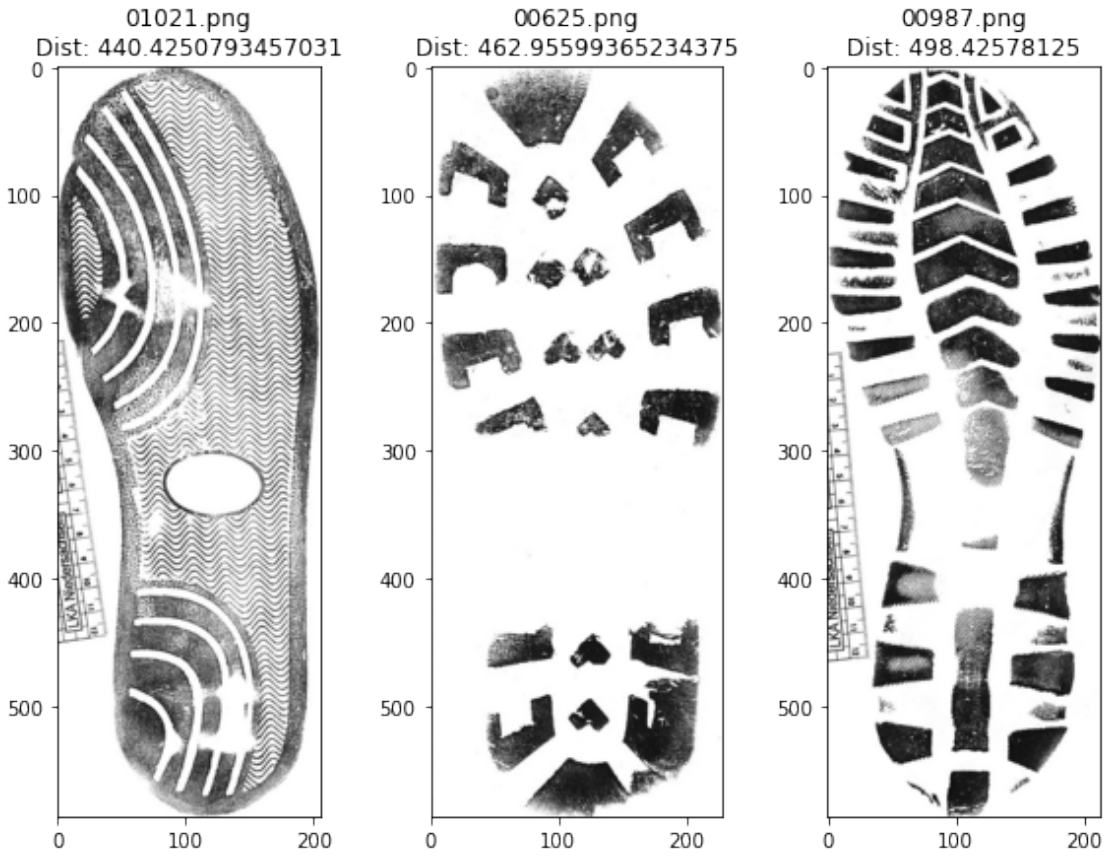
Less similar images



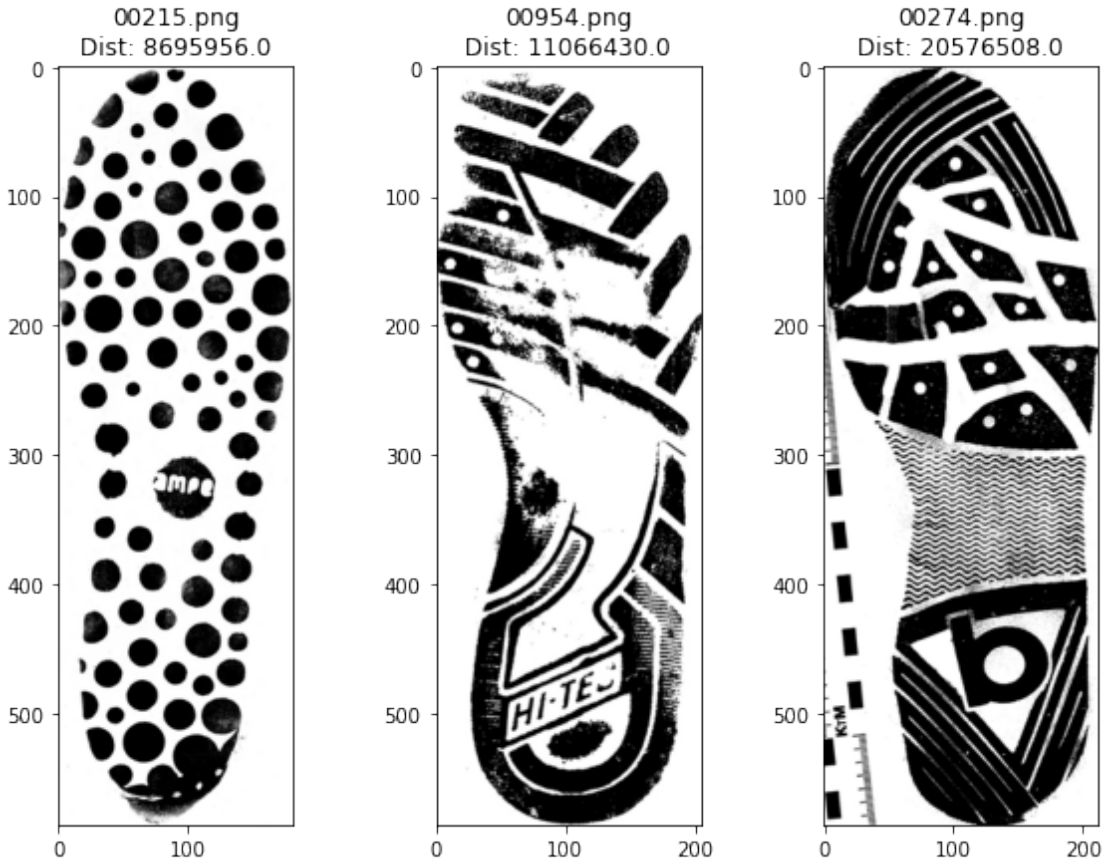
[69]:
```
#Test test0001 (converse)
test0001 = load_img('./fid300/tracks_cropped/{}'.format("00001.jpg"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test0001)
plt.show()
getTop3Similars(test0001)
```



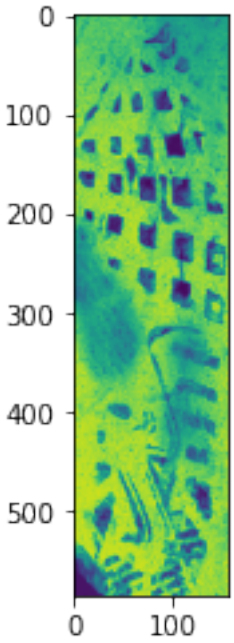

```
1/1 [==============================] - 0s 74ms/step
Most similar images
```



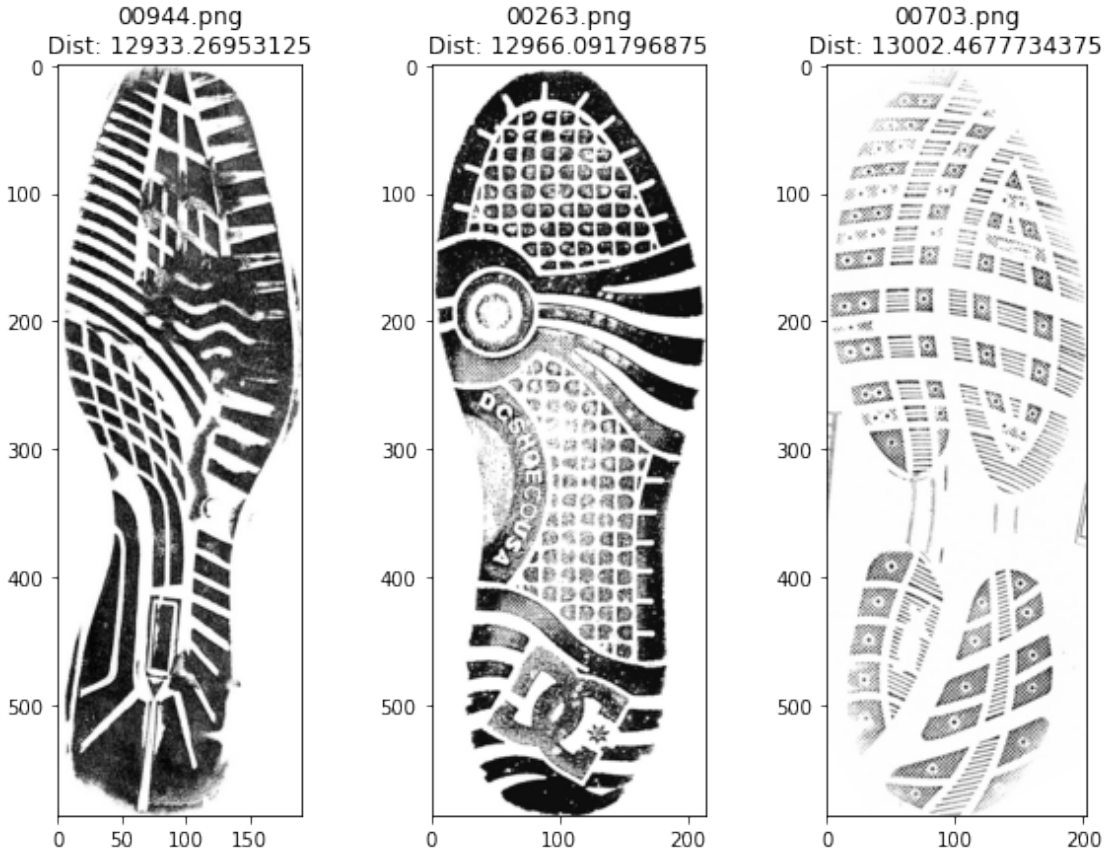
Less similar images



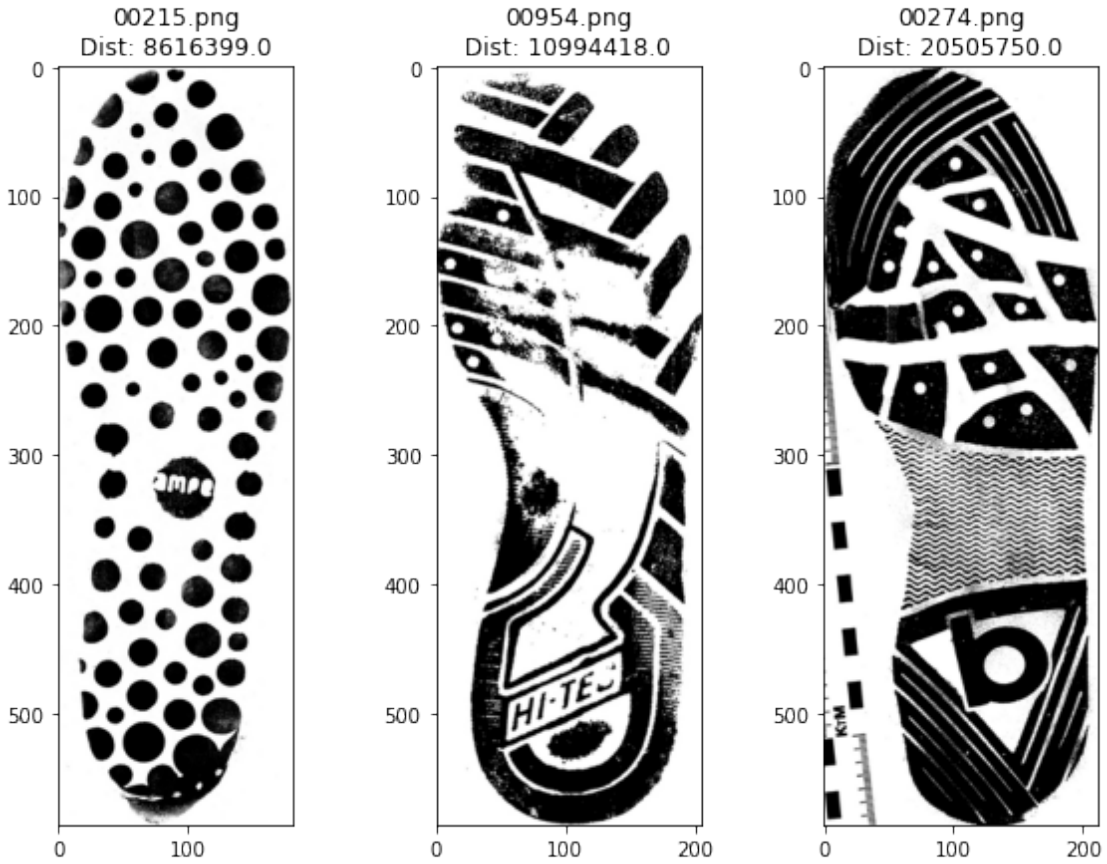
[70]:
```
#Test test0006 (huella entera)
test0006 = load_img('./fid300/tracks_cropped/{}'.format("00006.jpg"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test0006)
plt.show()
getTop3Similars(test0006)
```



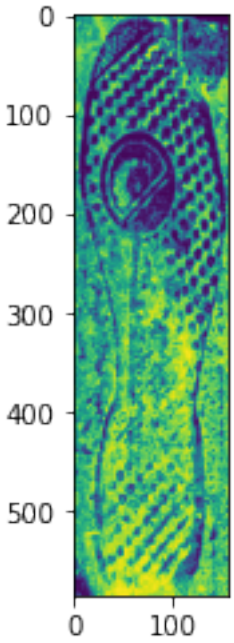

```
1/1 [==============================] - 0s 68ms/step
Most similar images
```



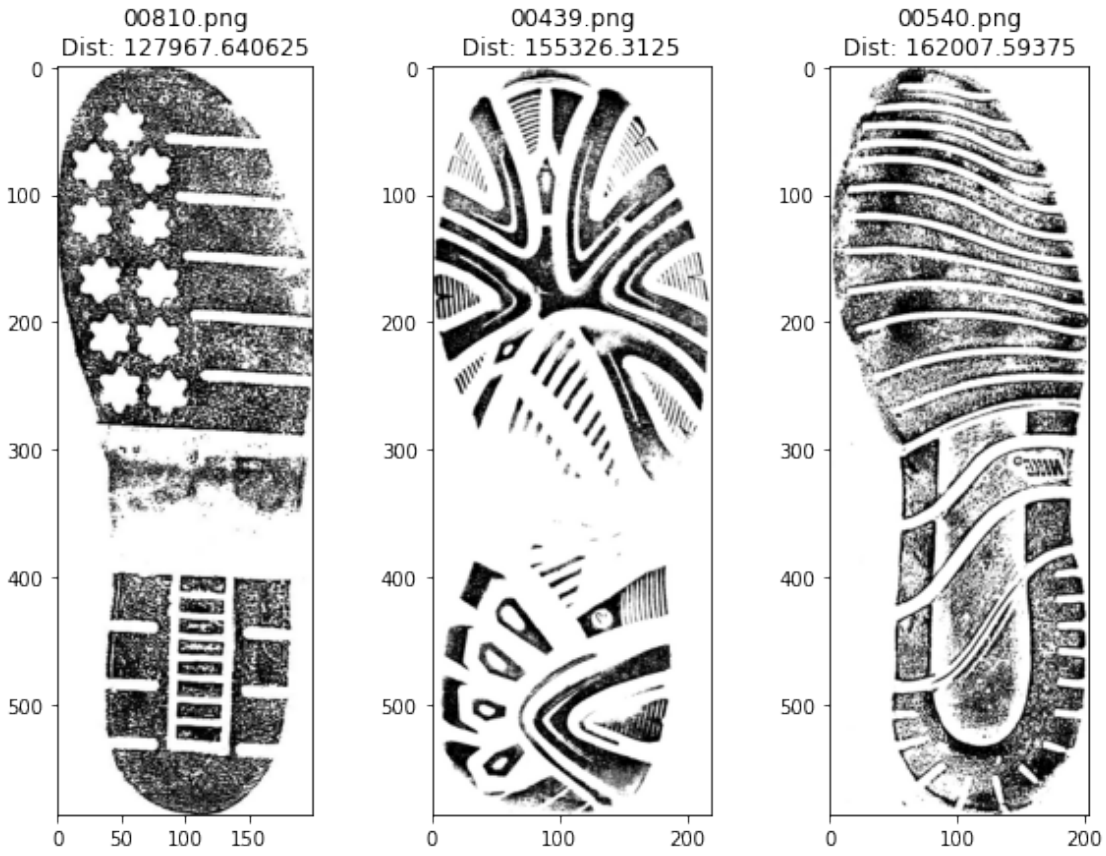
Less similar images



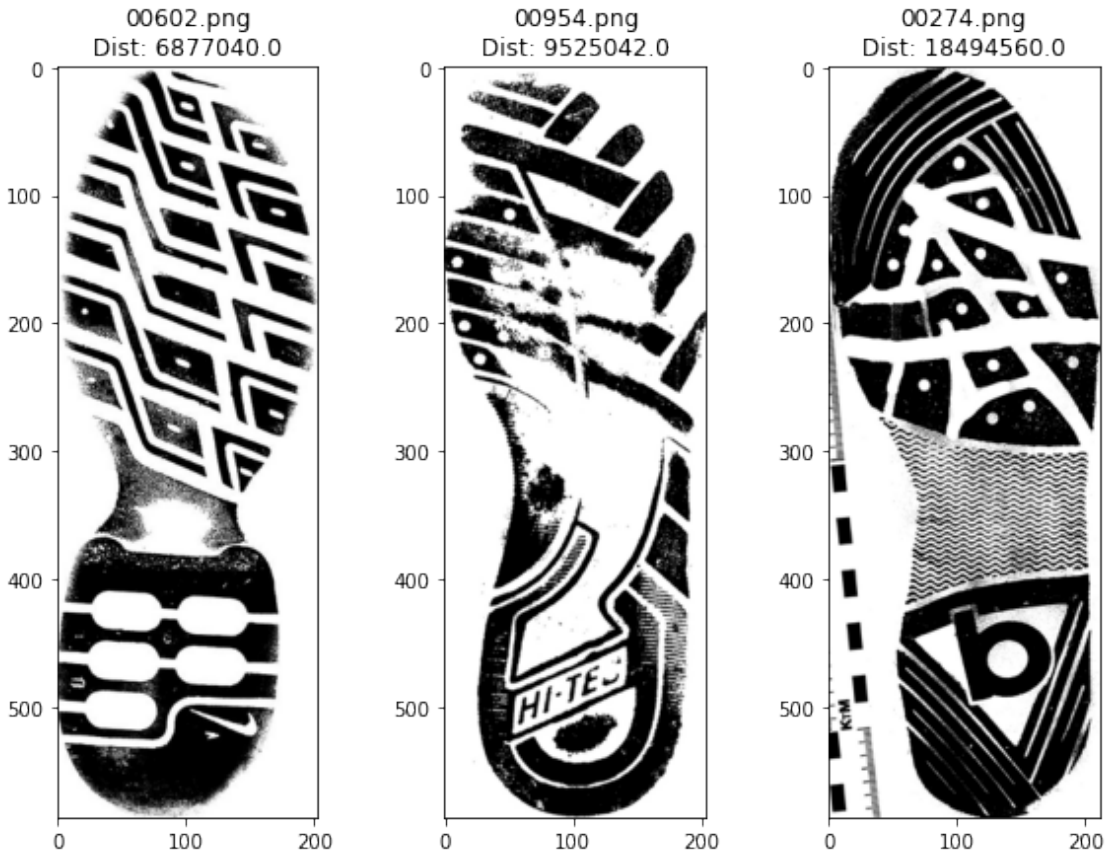
[71]:
```
#Test test0012 (huella entera)
test0012 = load_img('./fid300/tracks_cropped/{}'.format("00012.jpg"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test0012)
plt.show()
getTop3Similars(test0012)
```



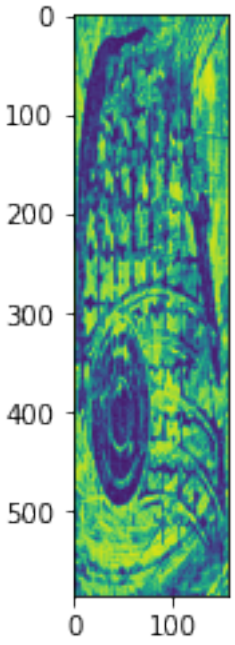

```
1/1 [==============================] - 0s 68ms/step
Most similar images
```



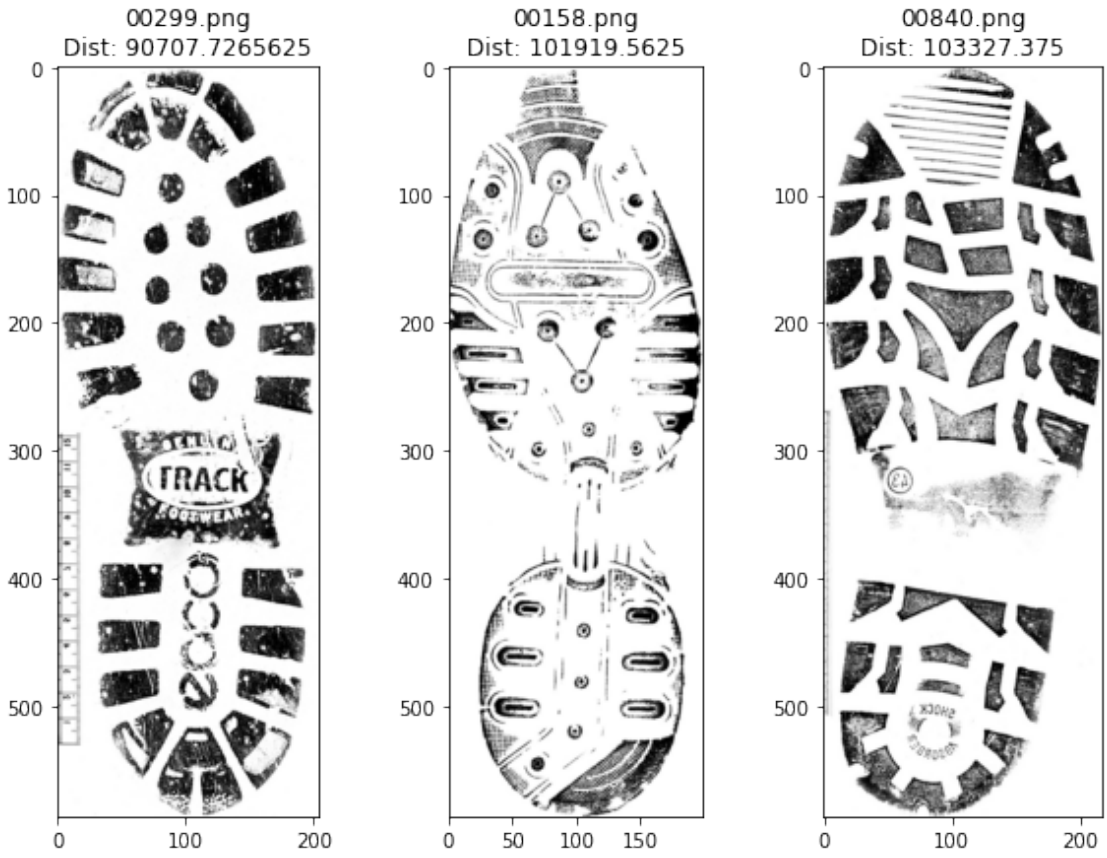
Less similar images



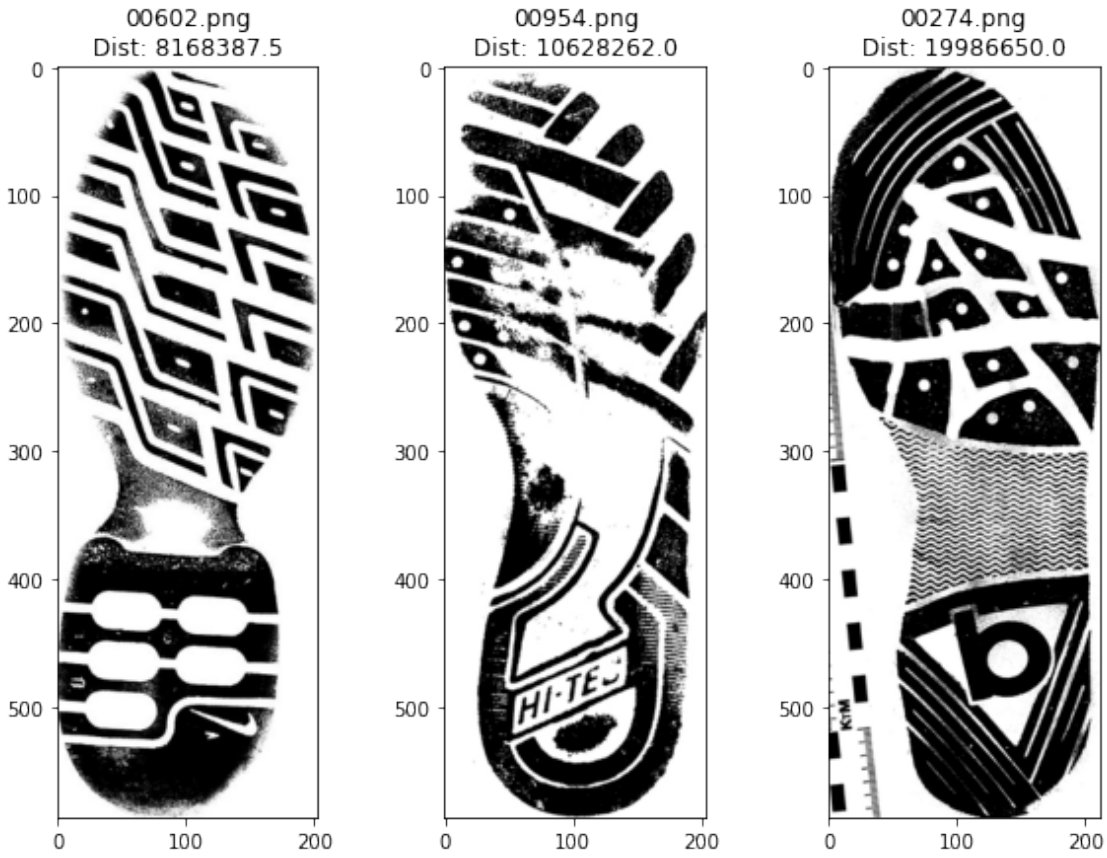
[72]:
```
#Test test0019 (huella mitad)
test0019 = load_img('./fid300/tracks_cropped/{}'.format("00019.jpg"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test0019)
plt.show()
getTop3Similars(test0019)
```



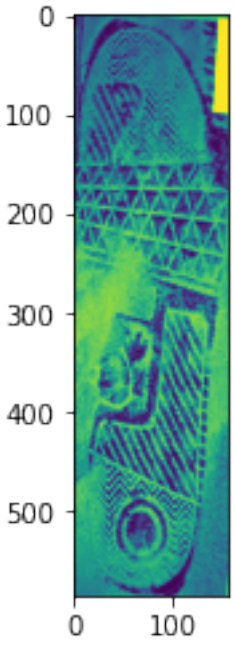

```
1/1 [==============================] - 0s 86ms/step
Most similar images
```



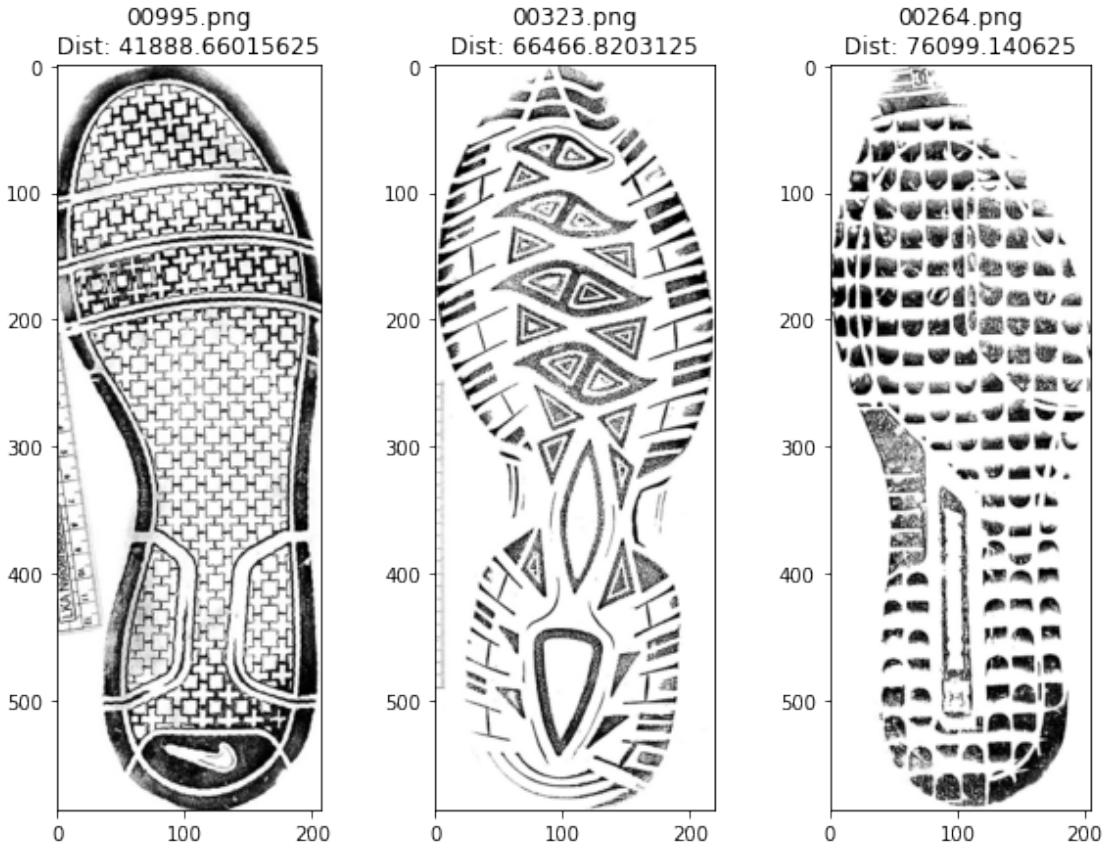
Less similar images



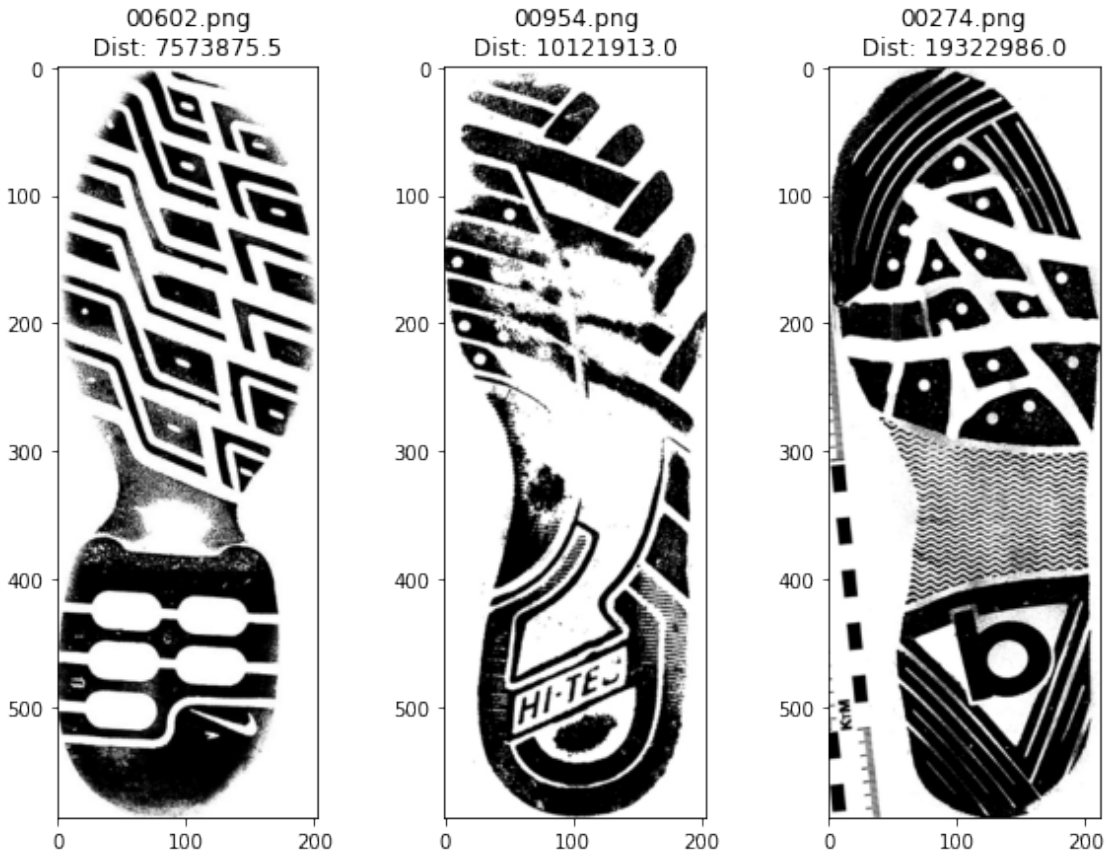
[74]:
```
#Test test0053 (huella más nítida)
test0053 = load_img('./fid300/tracks_cropped/{}'.format("00053.jpg"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test0053)
plt.show()
getTop3Similars(test0053)
```



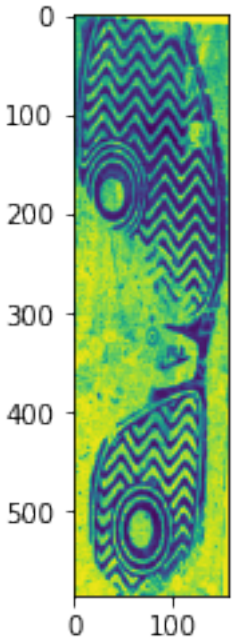

```
1/1 [==============================] - 0s 73ms/step
Most similar images
```



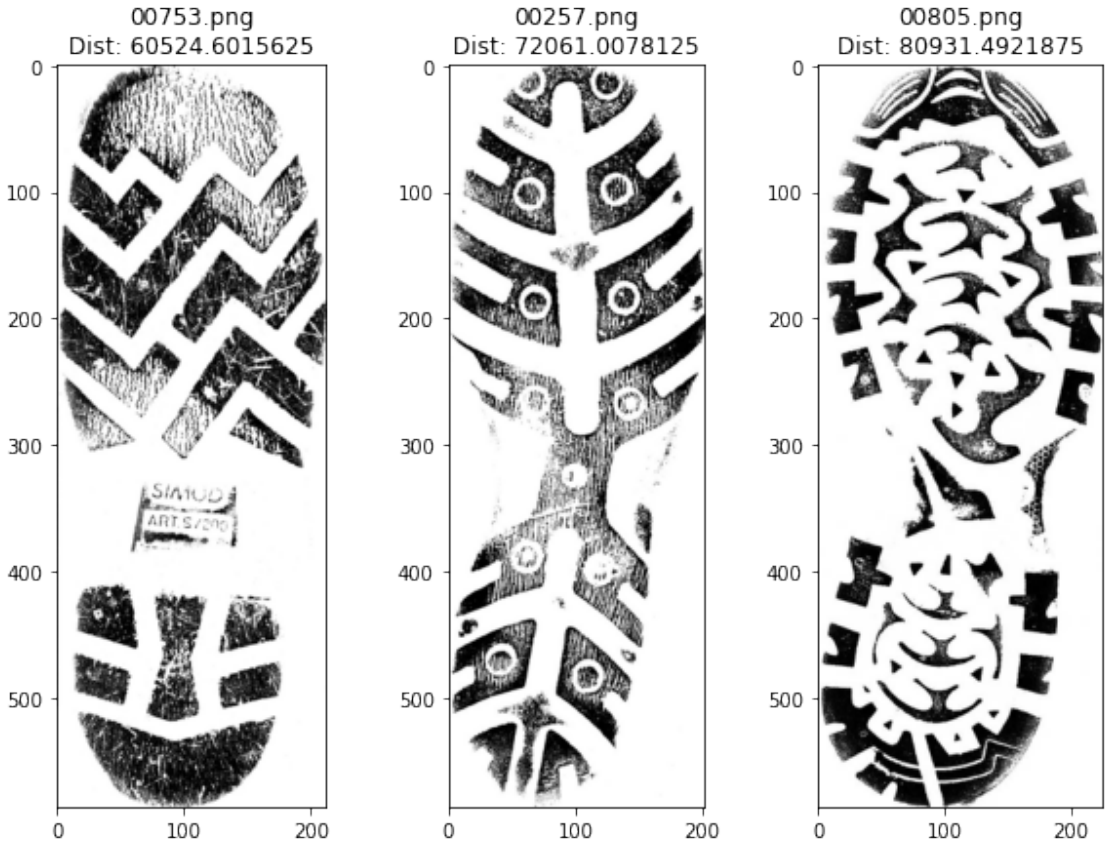
Less similar images



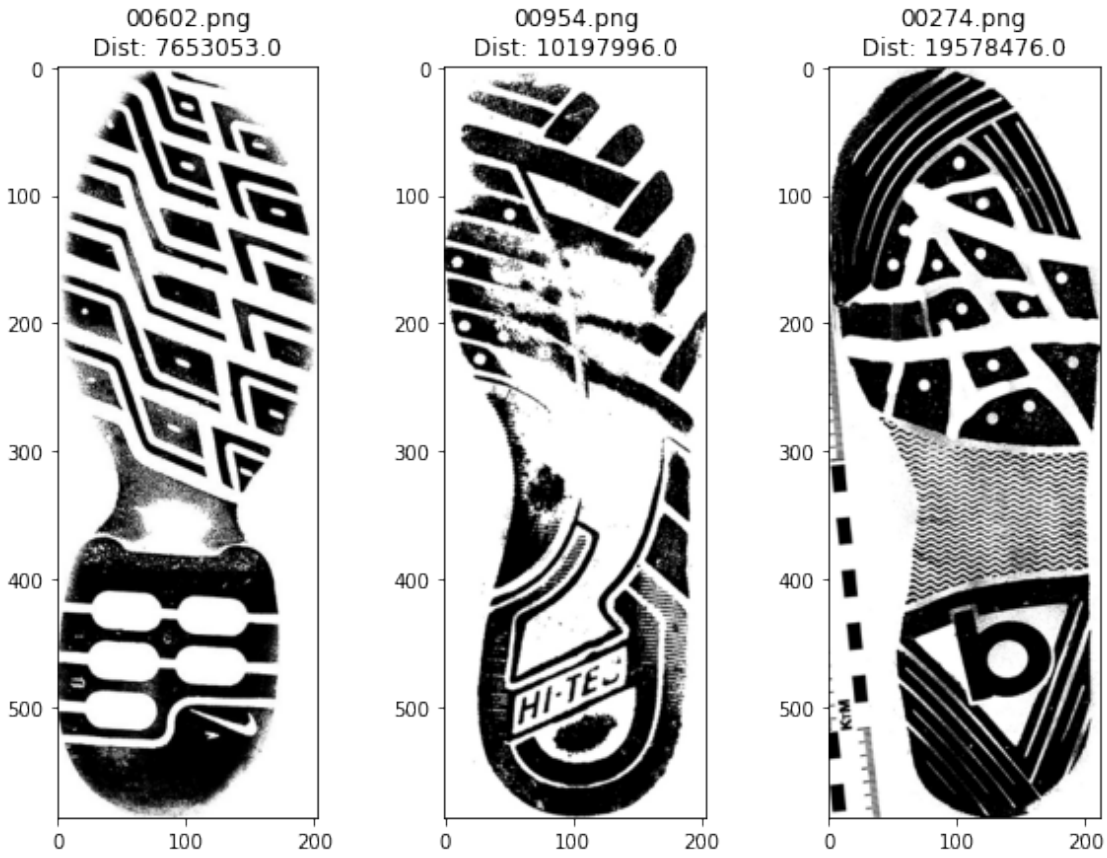
[76]:
```
#Test test0077 (huella nítida)
test0077 = load_img('./fid300/tracks_cropped/{}'.format("00077.jpg"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test0077)
plt.show()
getTop3Similars(test0077)
```



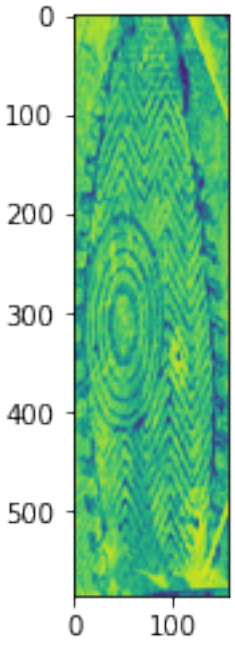

```
1/1 [==============================] - 0s 70ms/step
Most similar images
```



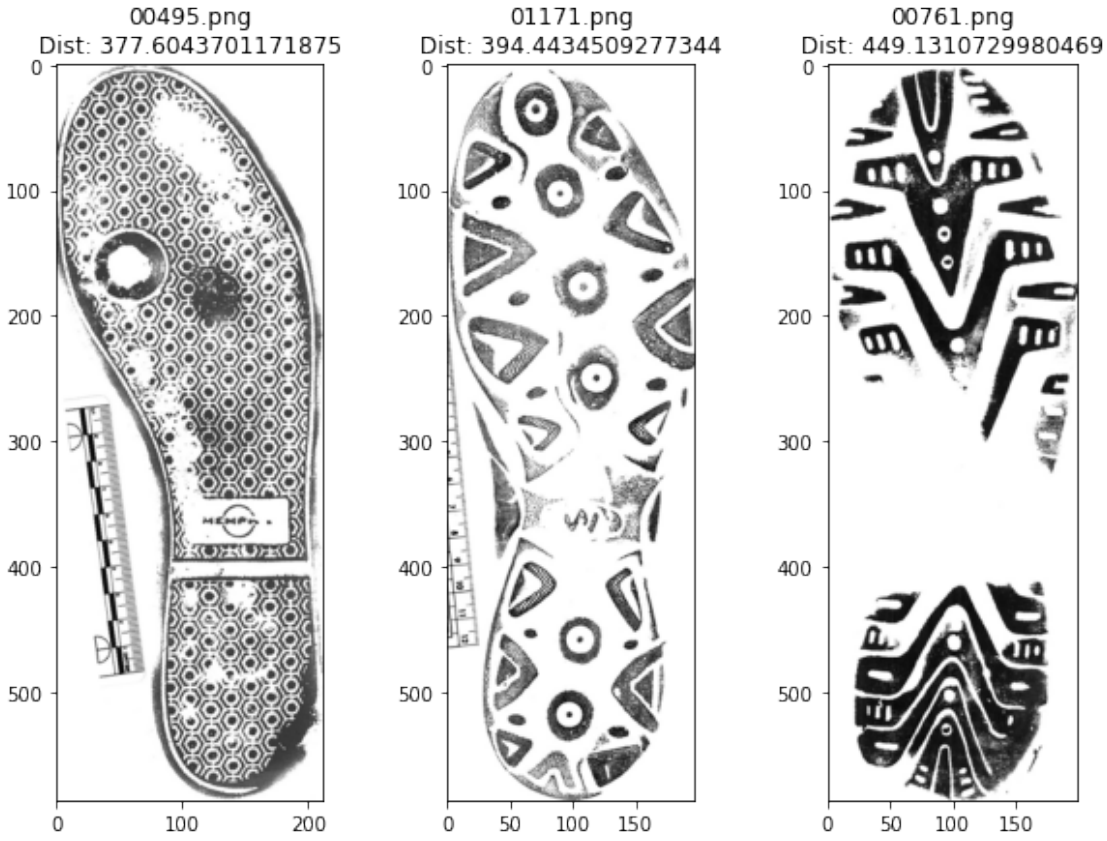
Less similar images



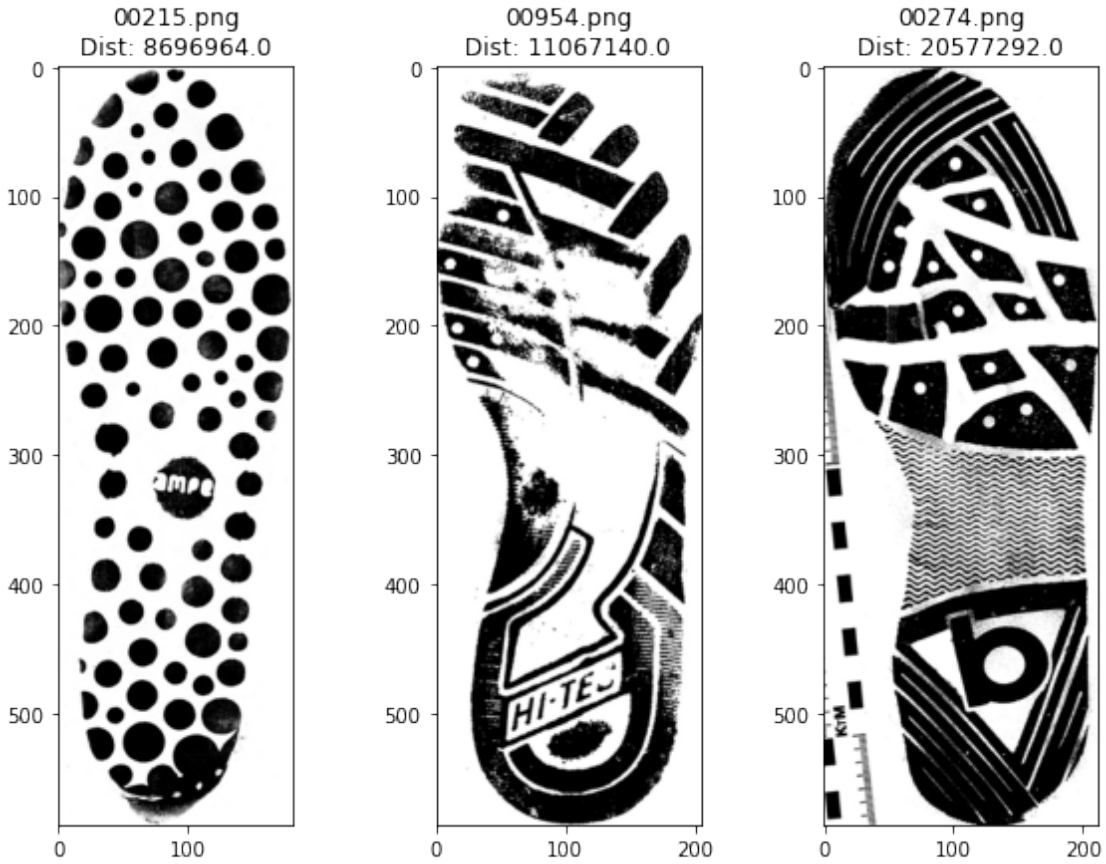
[80]:
```
#Test test0223 (huella media con un circulo)
test0223 = load_img('./fid300/tracks_cropped/{}'.format("00223.jpg"),
                    target_size = (img_height, img_width),␣
 ↪color_mode='grayscale')

plt.imshow(test0223)
plt.show()
getTop3Similars(test0223)
```




```
1/1 [==============================] - 0s 79ms/step
Most similar images
```



Less similar images



2 Analisis de resultados

En los experimentos con imágenes de referencia, se puede ver cómo en muchos de ellos las 3 imágenes más similares comparten alguna similitud. En cambio, en el caso de las menos similares, suelen aparecer las mismas en la mayoría de los casos (aunque mediante observación y bajo una representación subjetiva, no sean las menos parecidas).

Por último, en los experimentos con imágenes de huellas reales, en el caso de huellas enteras se puede observar alguna similitud como en *test0077*, pero en el caso de huellas parciales no, ya que en el caso del *test0223* se podría considerar aceptable ya que la huella más parecida contiene también un circulo a la izquierda, pero en otros casos, como en *test0001* o *test0019*, no obtiene un resultado satisfactorio.

Este cuaderno y modelo se ha creado como un ejercicio para aprender a crear y comprender una red siamesa, pero necesitaría más trabajo para que la búsqueda no fuera de la imágen entera, sino de fragmentos para mejorar la experiencia con huellas parciales ya que el modelo actual trata el fragmento como una huella entera.

[ ]: