21cs10052-code

November 18, 2023

1 Name: Pola Gnana Shekar

2 Roll No: 21CS10052

[ ]:

```
# if tensorflow is not preinstalled install by uncommenting the following␣
↪command
# pip install tensorflow
```

2.1 Data Importing and Preprocessing

[18]:

```
import os
import numpy as np
from PIL import Image
from tensorflow.keras.preprocessing.image import img_to_array, load_img
from tensorflow.keras.utils import to_categorical

# Define image directory
image_dir = 'Dataset2/FNA/benign/'
label = 1

# Initialize empty lists to store images and labels
bdata = []
blabels = []

# Load images and assign labels
for filename in os.listdir(image_dir):
    if filename.endswith(".png"):  # Check for image file extensions
        # Load image
        img = load_img(os.path.join(image_dir, filename), target_size=(224,␣
↪224))  # Adjust target_size as needed
        img_array = img_to_array(img)
        bdata.append(img_array)

        # Assign label
        blabels.append(label)

# Convert data and labels to numpy arrays
```

```python
bdata = np.array(bdata, dtype="float32")
blabels = np.array(blabels)

# Display of data and labels
print("Shape of bdata:", bdata.shape)
print("Pixel (RGB) data of: \n",bdata)
print("Labels:\n",blabels)
```

```
Shape of bdata: (1074, 224, 224, 3)
Pixel (RGB) data of:
 [[[[226. 164. 206.]
   [226. 164. 206.]
   [226. 164. 206.]
   …
   [243. 213. 235.]
   [243. 213. 235.]
   [243. 213. 235.]]

  [[226. 164. 206.]
   [226. 164. 206.]
   [226. 164. 206.]
   …
   [243. 213. 235.]
   [243. 213. 235.]
   [243. 213. 235.]]

  [[226. 164. 206.]
   [226. 164. 206.]
   [226. 164. 206.]
   …
   [243. 213. 235.]
   [243. 213. 235.]
   [243. 213. 235.]]

  …

  [[212. 125. 181.]
   [212. 125. 181.]
   [212. 125. 181.]
   …
   [215. 180. 211.]
   [215. 180. 211.]
   [215. 180. 211.]]

  [[212. 125. 181.]
   [212. 125. 181.]
   [212. 125. 181.]
   …
   [215. 180. 211.]
   [215. 180. 211.]
   [215. 180. 211.]]

  [[212. 125. 181.]
   [212. 125. 181.]
   [212. 125. 181.]
   …
   [215. 180. 211.]
   [215. 180. 211.]
   [215. 180. 211.]]]


 [[[219. 150. 197.]
   [219. 150. 197.]
   [219. 150. 197.]
   …
   [231. 193. 221.]
   [231. 193. 221.]
   [231. 193. 221.]]

  [[219. 150. 197.]
   [219. 150. 197.]
   [219. 150. 197.]
   …
   [231. 193. 221.]
   [231. 193. 221.]
   [231. 193. 221.]]

  [[219. 150. 197.]
   [219. 150. 197.]
   [219. 150. 197.]
   …
   [231. 193. 221.]
   [231. 193. 221.]
   [231. 193. 221.]]

  …

  [[230. 204. 226.]
   [230. 204. 226.]
   [230. 204. 226.]
   …
   [248. 247. 247.]
   [248. 247. 247.]
   [248. 247. 247.]]
  [[230. 204. 226.]
   [230. 204. 226.]
   [230. 204. 226.]
   …
   [248. 247. 247.]
   [248. 247. 247.]
   [248. 247. 247.]]

  [[230. 204. 226.]
   [230. 204. 226.]
   [230. 204. 226.]
   …
   [248. 247. 247.]
   [248. 247. 247.]
   [248. 247. 247.]]]


 [[[248. 245. 249.]
   [248. 245. 249.]
   [248. 245. 249.]
   …
   [248. 244. 248.]
   [248. 244. 248.]
   [248. 244. 248.]]

  [[248. 245. 249.]
   [248. 245. 249.]
   [248. 245. 249.]
   …
   [248. 244. 248.]
   [248. 244. 248.]
   [248. 244. 248.]]

  [[248. 245. 249.]
   [248. 245. 249.]
   [248. 245. 249.]
   …
   [248. 244. 248.]
   [248. 244. 248.]
   [248. 244. 248.]]

  …

  [[249. 247. 249.]
   [249. 247. 249.]
   [249. 247. 249.]
   …
   [186. 120. 177.]
   [186. 120. 177.]
   [186. 120. 177.]]

  [[249. 247. 249.]
   [249. 247. 249.]
   [249. 247. 249.]
   …
   [186. 120. 177.]
   [186. 120. 177.]
   [186. 120. 177.]]

  [[249. 247. 249.]
   [249. 247. 249.]
   [249. 247. 249.]
   …
   [186. 120. 177.]
   [186. 120. 177.]
   [186. 120. 177.]]]


 …


 [[[242. 240. 244.]
   [242. 240. 244.]
   [242. 240. 244.]
   …
   [243. 240. 244.]
   [243. 240. 244.]
   [243. 240. 244.]]

  [[242. 240. 244.]
   [242. 240. 244.]
   [242. 240. 244.]
   …
   [243. 240. 244.]
   [243. 240. 244.]
   [243. 240. 244.]]

  [[242. 240. 244.]
   [242. 240. 244.]
   [242. 240. 244.]
   …
   [243. 240. 244.]
   [243. 240. 244.]
   [243. 240. 244.]]

  …
  [[231. 216. 225.]
   [231. 216. 225.]
   [231. 216. 225.]
   …
   [241. 240. 241.]
   [241. 240. 241.]
   [241. 240. 241.]]

  [[231. 216. 225.]
   [231. 216. 225.]
   [231. 216. 225.]
   …
   [241. 240. 241.]
   [241. 240. 241.]
   [241. 240. 241.]]

  [[231. 216. 225.]
   [231. 216. 225.]
   [231. 216. 225.]
   …
   [241. 240. 241.]
   [241. 240. 241.]
   [241. 240. 241.]]]


 [[[241. 239. 243.]
   [241. 239. 243.]
   [241. 239. 243.]
   …
   [244. 239. 237.]
   [244. 239. 237.]
   [244. 239. 237.]]

  [[241. 239. 243.]
   [241. 239. 243.]
   [241. 239. 243.]
   …
   [244. 239. 237.]
   [244. 239. 237.]
   [244. 239. 237.]]

  [[241. 239. 243.]
   [241. 239. 243.]
   [241. 239. 243.]
   …
   [244. 239. 237.]
   [244. 239. 237.]
   [244. 239. 237.]]

  …

  [[172.  93. 143.]
   [172.  93. 143.]
   [172.  93. 143.]
   …
   [180. 145. 177.]
   [180. 145. 177.]
   [180. 145. 177.]]

  [[172.  93. 143.]
   [172.  93. 143.]
   [172.  93. 143.]
   …
   [180. 145. 177.]
   [180. 145. 177.]
   [180. 145. 177.]]

  [[172.  93. 143.]
   [172.  93. 143.]
   [172.  93. 143.]
   …
   [180. 145. 177.]
   [180. 145. 177.]
   [180. 145. 177.]]]


 [[[177.  99. 150.]
   [177.  99. 150.]
   [177.  99. 150.]
   …
   [242. 233. 239.]
   [242. 233. 239.]
   [242. 233. 239.]]

  [[177.  99. 150.]
   [177.  99. 150.]
   [177.  99. 150.]
   …
   [242. 233. 239.]
   [242. 233. 239.]
   [242. 233. 239.]]

  [[177.  99. 150.]
   [177.  99. 150.]
   [177.  99. 150.]
```

```
   …
   [242. 233. 239.]
   [242. 233. 239.]
   [242. 233. 239.]]

  …

  [[237. 235. 239.]
   [237. 235. 239.]
   [237. 235. 239.]
   …
   [194. 114. 164.]
   [194. 114. 164.]
   [194. 114. 164.]]

  [[237. 235. 239.]
   [237. 235. 239.]
   [237. 235. 239.]
   …
   [194. 114. 164.]
   [194. 114. 164.]
   [194. 114. 164.]]

  [[237. 235. 239.]
   [237. 235. 239.]
   [237. 235. 239.]
   …
   [194. 114. 164.]
   [194. 114. 164.]
   [194. 114. 164.]]]]
Labels:
 [1 1 1 … 1 1 1]
```

[3]:

```python
import os
import numpy as np
from PIL import Image
from tensorflow.keras.preprocessing.image import img_to_array, load_img
from tensorflow.keras.utils import to_categorical

# Define image directory
image_dir = 'Dataset2/FNA/malignant/'
label = 0

# Initialize empty lists to store images and labels
mdata = []
mlabels = []
```

```python
# Load images and assign labels
for filename in os.listdir(image_dir):
    if filename.endswith(".png"):  # Check for image file extensions
        # Load image
        img = load_img(os.path.join(image_dir, filename), target_size=(224, 224))  # Adjust target_size as needed
        img_array = img_to_array(img)
        mdata.append(img_array)

        # Assign label
        mlabels.append(label)

# Convert data and labels to numpy arrays
mdata = np.array(mdata, dtype="float32")
mlabels = np.array(mlabels)

# Display of data and labels
print("Shape of mdata:", mdata.shape)
print("Pixel (RGB) data of: \n",mdata)
print("Labels:\n",mlabels)
```

```
Shape of mdata: (650, 224, 224, 3)
Pixel (RGB) data of:
 [[[[252. 253. 253.]
   [252. 253. 253.]
   [252. 253. 253.]
   …
   [251. 251. 252.]
   [251. 251. 252.]
   [251. 251. 252.]]

  [[252. 253. 253.]
   [252. 253. 253.]
   [252. 253. 253.]
   …
   [251. 251. 252.]
   [251. 251. 252.]
   [251. 251. 252.]]

  [[252. 253. 253.]
   [252. 253. 253.]
   [252. 253. 253.]
   …
   [251. 251. 252.]
   [251. 251. 252.]
   [251. 251. 252.]]
  …

  [[252. 252. 252.]
   [252. 252. 252.]
   [252. 252. 252.]
   …
   [138.  99. 154.]
   [138.  99. 154.]
   [138.  99. 154.]]

  [[252. 252. 252.]
   [252. 252. 252.]
   [252. 252. 252.]
   …
   [138.  99. 154.]
   [138.  99. 154.]
   [138.  99. 154.]]

  [[252. 252. 252.]
   [252. 252. 252.]
   [252. 252. 252.]
   …
   [138.  99. 154.]
   [138.  99. 154.]
   [138.  99. 154.]]]


 [[[251. 252. 252.]
   [251. 252. 252.]
   [251. 252. 252.]
   …
   [128.  81. 146.]
   [128.  81. 146.]
   [128.  81. 146.]]

  [[251. 252. 252.]
   [251. 252. 252.]
   [251. 252. 252.]
   …
   [128.  81. 146.]
   [128.  81. 146.]
   [128.  81. 146.]]

  [[251. 252. 252.]
   [251. 252. 252.]
   [251. 252. 252.]
   …
   [128.  81. 146.]
   [128.  81. 146.]
   [128.  81. 146.]]

  …

  [[221. 181. 219.]
   [221. 181. 219.]
   [221. 181. 219.]
   …
   [214. 201. 227.]
   [214. 201. 227.]
   [214. 201. 227.]]

  [[221. 181. 219.]
   [221. 181. 219.]
   [221. 181. 219.]
   …
   [214. 201. 227.]
   [214. 201. 227.]
   [214. 201. 227.]]

  [[221. 181. 219.]
   [221. 181. 219.]
   [221. 181. 219.]
   …
   [214. 201. 227.]
   [214. 201. 227.]
   [214. 201. 227.]]]


 [[[252. 252. 252.]
   [252. 252. 252.]
   [252. 252. 252.]
   …
   [ 92.  46. 107.]
   [ 92.  46. 107.]
   [ 92.  46. 107.]]

  [[252. 252. 252.]
   [252. 252. 252.]
   [252. 252. 252.]
   …
   [ 92.  46. 107.]
   [ 92.  46. 107.]
   [ 92.  46. 107.]]

  [[252. 252. 252.]
   [252. 252. 252.]
   [252. 252. 252.]
   …
   [ 92.  46. 107.]
   [ 92.  46. 107.]
   [ 92.  46. 107.]]

  …

  [[251. 251. 252.]
   [251. 251. 252.]
   [251. 251. 252.]
   …
   [249. 247. 251.]
   [249. 247. 251.]
   [249. 247. 251.]]

  [[251. 251. 252.]
   [251. 251. 252.]
   [251. 251. 252.]
   …
   [249. 247. 251.]
   [249. 247. 251.]
   [249. 247. 251.]]

  [[251. 251. 252.]
   [251. 251. 252.]
   [251. 251. 252.]
   …
   [249. 247. 251.]
   [249. 247. 251.]
   [249. 247. 251.]]]


 …


 [[[141.  66. 111.]
   [141.  66. 111.]
   [141.  66. 111.]
   …
   [234. 221. 229.]
   [234. 221. 229.]
   [234. 221. 229.]]

  [[141.  66. 111.]
   [141.  66. 111.]
   [141.  66. 111.]
   …
   [234. 221. 229.]
   [234. 221. 229.]
   [234. 221. 229.]]

  [[141.  66. 111.]
   [141.  66. 111.]
   [141.  66. 111.]
   …
   [234. 221. 229.]
   [234. 221. 229.]
   [234. 221. 229.]]

  …

  [[186.  92. 132.]
   [186.  92. 132.]
   [186.  92. 132.]
   …
   [136.  79. 126.]
   [136.  79. 126.]
   [136.  79. 126.]]

  [[186.  92. 132.]
   [186.  92. 132.]
   [186.  92. 132.]
   …
   [136.  79. 126.]
   [136.  79. 126.]
   [136.  79. 126.]]

  [[186.  92. 132.]
   [186.  92. 132.]
   [186.  92. 132.]
   …
   [136.  79. 126.]
   [136.  79. 126.]
   [136.  79. 126.]]]


 [[[151.  68. 116.]
   [151.  68. 116.]
   [151.  68. 116.]
   …
   [122.  74. 127.]
   [122.  74. 127.]
   [122.  74. 127.]]

  [[151.  68. 116.]
```

```
  [151.  68. 116.]
  [151.  68. 116.]
  …
  [122.  74. 127.]
  [122.  74. 127.]
  [122.  74. 127.]]

 [[151.  68. 116.]
  [151.  68. 116.]
  [151.  68. 116.]
  …
  [122.  74. 127.]
  [122.  74. 127.]
  [122.  74. 127.]]

 …

 [[145.  98. 148.]
  [145.  98. 148.]
  [145.  98. 148.]
  …
  [169.  76. 122.]
  [169.  76. 122.]
  [169.  76. 122.]]

 [[145.  98. 148.]
  [145.  98. 148.]
  [145.  98. 148.]
  …
  [169.  76. 122.]
  [169.  76. 122.]
  [169.  76. 122.]]

 [[145.  98. 148.]
  [145.  98. 148.]
  [145.  98. 148.]
  …
  [169.  76. 122.]
  [169.  76. 122.]
  [169.  76. 122.]]]


[[[141.  96. 147.]
  [141.  96. 147.]
  [141.  96. 147.]
  …
  [143.  74. 126.]
  [143.  74. 126.]
```

```
   [143.  74. 126.]]

  [[141.  96. 147.]
   [141.  96. 147.]
   [141.  96. 147.]
   …
   [143.  74. 126.]
   [143.  74. 126.]
   [143.  74. 126.]]

  [[141.  96. 147.]
   [141.  96. 147.]
   [141.  96. 147.]
   …
   [143.  74. 126.]
   [143.  74. 126.]
   [143.  74. 126.]]

  …

  [[113.  63. 118.]
   [113.  63. 118.]
   [113.  63. 118.]
   …
   [176. 143. 177.]
   [176. 143. 177.]
   [176. 143. 177.]]

  [[113.  63. 118.]
   [113.  63. 118.]
   [113.  63. 118.]
   …
   [176. 143. 177.]
   [176. 143. 177.]
   [176. 143. 177.]]

  [[113.  63. 118.]
   [113.  63. 118.]
   [113.  63. 118.]
   …
   [176. 143. 177.]
   [176. 143. 177.]
   [176. 143. 177.]]]]
Labels:
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

[4]:

```python
import os
import numpy as np
from PIL import Image
from tensorflow.keras.preprocessing.image import img_to_array, load_img

# Define image directory for unlabeled images
unlabeled_image_dir = "Dataset2/test/"

# Initialize an empty list to store unlabeled images
unlabeled_data = []

# Load unlabeled images
for filename in os.listdir(unlabeled_image_dir):
    if filename.endswith(".jpg") or filename.endswith(".png"):  # Check for␣
 ↪image file extensions
        # Load image
        img = load_img(os.path.join(unlabeled_image_dir, filename),␣
 ↪target_size=(224, 224))  # Adjust target_size as needed
        img_array = img_to_array(img)
        unlabeled_data.append(img_array)

# Convert data to a numpy array
unlabeled_data = np.array(unlabeled_data, dtype="float32")

# Display shape of the unlabeled data
print("Shape of unlabeled data:", unlabeled_data.shape)
print(unlabeled_data)
```

```
Shape of unlabeled data: (14, 224, 224, 3)
[[[[144.  93. 149.]
   [144.  93. 149.]
   [144.  93. 149.]
   …
```

```
  [117.  79. 126.]
  [117.  79. 126.]
  [117.  79. 126.]]

 [[144.  93. 149.]
  [144.  93. 149.]
  [144.  93. 149.]
  …
  [117.  79. 126.]
  [117.  79. 126.]
  [117.  79. 126.]]

 [[144.  93. 149.]
  [144.  93. 149.]
  [144.  93. 149.]
  …
  [117.  79. 126.]
  [117.  79. 126.]
  [117.  79. 126.]]

 …

 [[199. 126. 171.]
  [199. 126. 171.]
  [199. 126. 171.]
  …
  [237. 234. 237.]
  [237. 234. 237.]
  [237. 234. 237.]]

 [[199. 126. 171.]
  [199. 126. 171.]
  [199. 126. 171.]
  …
  [237. 234. 237.]
  [237. 234. 237.]
  [237. 234. 237.]]

 [[199. 126. 171.]
  [199. 126. 171.]
  [199. 126. 171.]
  …
  [237. 234. 237.]
  [237. 234. 237.]
  [237. 234. 237.]]]


[[[ 88.  50. 113.]
```

```
 [ 88.  50. 113.]
 [ 88.  50. 113.]
 …
 [181. 132. 179.]
 [181. 132. 179.]
 [181. 132. 179.]]

[[ 88.  50. 113.]
 [ 88.  50. 113.]
 [ 88.  50. 113.]
 …
 [181. 132. 179.]
 [181. 132. 179.]
 [181. 132. 179.]]

[[ 88.  50. 113.]
 [ 88.  50. 113.]
 [ 88.  50. 113.]
 …
 [181. 132. 179.]
 [181. 132. 179.]
 [181. 132. 179.]]

…

[[139.  84. 149.]
 [139.  84. 149.]
 [139.  84. 149.]
 …
 [135. 100. 149.]
 [135. 100. 149.]
 [135. 100. 149.]]

[[139.  84. 149.]
 [139.  84. 149.]
 [139.  84. 149.]
 …
 [135. 100. 149.]
 [135. 100. 149.]
 [135. 100. 149.]]

[[139.  84. 149.]
 [139.  84. 149.]
 [139.  84. 149.]
 …
 [135. 100. 149.]
 [135. 100. 149.]
 [135. 100. 149.]]]
```

```
[[[207. 103. 159.]
  [207. 103. 159.]
  [207. 103. 159.]
  …
  [205.  79. 145.]
  [205.  79. 145.]
  [205.  79. 145.]]

 [[207. 103. 159.]
  [207. 103. 159.]
  [207. 103. 159.]
  …
  [205.  79. 145.]
  [205.  79. 145.]
  [205.  79. 145.]]

 [[207. 103. 159.]
  [207. 103. 159.]
  [207. 103. 159.]
  …
  [205.  79. 145.]
  [205.  79. 145.]
  [205.  79. 145.]]

 …

 [[226. 153. 198.]
  [226. 153. 198.]
  [226. 153. 198.]
  …
  [249. 247. 251.]
  [249. 247. 251.]
  [249. 247. 251.]]

 [[226. 153. 198.]
  [226. 153. 198.]
  [226. 153. 198.]
  …
  [249. 247. 251.]
  [249. 247. 251.]
  [249. 247. 251.]]

 [[226. 153. 198.]
  [226. 153. 198.]
  [226. 153. 198.]
  …
  [249. 247. 251.]
  [249. 247. 251.]
  [249. 247. 251.]]]


…


[[[158. 115. 149.]
  [158. 115. 149.]
  [158. 115. 149.]
  …
  [198. 136. 183.]
  [198. 136. 183.]
  [198. 136. 183.]]

 [[158. 115. 149.]
  [158. 115. 149.]
  [158. 115. 149.]
  …
  [198. 136. 183.]
  [198. 136. 183.]
  [198. 136. 183.]]

 [[158. 115. 149.]
  [158. 115. 149.]
  [158. 115. 149.]
  …
  [198. 136. 183.]
  [198. 136. 183.]
  [198. 136. 183.]]

 …

 [[208.  88. 138.]
  [208.  88. 138.]
  [208.  88. 138.]
  …
  [158.  87. 146.]
  [158.  87. 146.]
  [158.  87. 146.]]

 [[208.  88. 138.]
  [208.  88. 138.]
  [208.  88. 138.]
  …
  [158.  87. 146.]
  [158.  87. 146.]
  [158.  87. 146.]]

 [[208.  88. 138.]
  [208.  88. 138.]
  [208.  88. 138.]
  …
  [158.  87. 146.]
  [158.  87. 146.]
  [158.  87. 146.]]]


[[[219. 159. 201.]
  [219. 159. 201.]
  [219. 159. 201.]
  …
  [214. 168. 209.]
  [214. 168. 209.]
  [214. 168. 209.]]

 [[219. 159. 201.]
  [219. 159. 201.]
  [219. 159. 201.]
  …
  [214. 168. 209.]
  [214. 168. 209.]
  [214. 168. 209.]]

 [[219. 159. 201.]
  [219. 159. 201.]
  [219. 159. 201.]
  …
  [214. 168. 209.]
  [214. 168. 209.]
  [214. 168. 209.]]

 …

 [[248. 244. 247.]
  [248. 244. 247.]
  [248. 244. 247.]
  …
  [245. 245. 246.]
  [245. 245. 246.]
  [245. 245. 246.]]

 [[248. 244. 247.]
  [248. 244. 247.]
  [248. 244. 247.]
  …
  [245. 245. 246.]
  [245. 245. 246.]
  [245. 245. 246.]]

 [[248. 244. 247.]
  [248. 244. 247.]
  [248. 244. 247.]
  …
  [245. 245. 246.]
  [245. 245. 246.]
  [245. 245. 246.]]]


[[[238. 217. 235.]
  [238. 217. 235.]
  [238. 217. 235.]
  …
  [252. 250. 251.]
  [252. 250. 251.]
  [252. 250. 251.]]

 [[238. 217. 235.]
  [238. 217. 235.]
  [238. 217. 235.]
  …
  [252. 250. 251.]
  [252. 250. 251.]
  [252. 250. 251.]]

 [[238. 217. 235.]
  [238. 217. 235.]
  [238. 217. 235.]
  …
  [252. 250. 251.]
  [252. 250. 251.]
  [252. 250. 251.]]

 …

 [[246. 241. 245.]
  [246. 241. 245.]
  [246. 241. 245.]
  …
  [221. 128. 178.]
  [221. 128. 178.]
  [221. 128. 178.]]
 [[246. 241. 245.]
  [246. 241. 245.]
  [246. 241. 245.]
  …
  [221. 128. 178.]
  [221. 128. 178.]
  [221. 128. 178.]]

 [[246. 241. 245.]
  [246. 241. 245.]
  [246. 241. 245.]
  …
  [221. 128. 178.]
  [221. 128. 178.]
  [221. 128. 178.]]]]
```

[5]:

```python
import numpy as np

# Function to normalize pixel values between 0 and 1 by dividing by 255
def normalize_images(images):
    normalized_data = images / 255.0  # Normalize pixel values between 0 and 1
    return normalized_data

# Function to convert normalized values to 0 or 1 based on a threshold
def round_threshold(images):
    processed_data = np.where(images <= 0.5, 0, 1)  # Round values > 0.5 to 1,␣
 ↪else to 0
    return processed_data

# Normalize pixel values into 0 and 1
normalized_bdata = normalize_images(bdata)
processed_bdata= round_threshold(normalized_bdata)

normalized_mdata = normalize_images(mdata)
processed_mdata= round_threshold(normalized_mdata)

normalized_data = normalize_images(unlabeled_data)
processed_data = round_threshold(normalized_data)
```

[6]:

```python
# Concatenate the data and labels
X = np.concatenate((processed_bdata, processed_mdata), axis=0)
y = np.concatenate((blabels, mlabels), axis=0)

# Display shapes of the combined dataset and labels
print("Shape of combined data:", X.shape)
print("Shape of combined labels:", y.shape)
```

```
Shape of combined data: (1724, 224, 224, 3)
Shape of combined labels: (1724,)
```

[7]:

```python
from sklearn.model_selection import train_test_split

# Split the data into training and validation sets (adjust test_size and random_state as needed)
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)

# Display the shapes of the train and validation sets
print("Shape of X_train:", X_train.shape)
print("Shape of X_val:", X_val.shape)
print("Shape of y_train:", y_train.shape)
print("Shape of y_val:", y_val.shape)
```

```
Shape of X_train: (1379, 224, 224, 3)
Shape of X_val: (345, 224, 224, 3)
Shape of y_train: (1379,)
Shape of y_val: (345,)
```

2.2 Implementation of CNN model

[17]:

```python
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.losses import SparseCategoricalCrossentropy

import warnings

# Ignore deprecation warnings related to a specific function or module
warnings.filterwarnings('ignore')

# Define the CNN model architecture
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3)),
    MaxPooling2D((2, 2)),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(1, activation='sigmoid')
])
```

```
# Compile the model
model.compile(optimizer=tf.keras.optimizers.Adam(), loss=tf.keras.losses.
 ↪BinaryCrossentropy(), metrics=[tf.keras.metrics.BinaryAccuracy()])

# Train the model using the training set and validate using the validation set
history = model.fit(X_train, y_train, epochs=10, batch_size=32,␣
 ↪validation_data=(X_val, y_val))
```

```
Epoch 1/10
44/44 [==============================] - 55s 1s/step - loss: 0.5476 -
binary_accuracy: 0.8535 - val_loss: 0.2467 - val_binary_accuracy: 0.9275
Epoch 2/10
44/44 [==============================] - 59s 1s/step - loss: 0.2500 -
binary_accuracy: 0.9173 - val_loss: 0.2285 - val_binary_accuracy: 0.9304
Epoch 3/10
44/44 [==============================] - 58s 1s/step - loss: 0.2253 -
binary_accuracy: 0.9260 - val_loss: 0.2533 - val_binary_accuracy: 0.9130
Epoch 4/10
44/44 [==============================] - 58s 1s/step - loss: 0.2072 -
binary_accuracy: 0.9253 - val_loss: 0.1992 - val_binary_accuracy: 0.9420
Epoch 5/10
44/44 [==============================] - 60s 1s/step - loss: 0.2015 -
binary_accuracy: 0.9275 - val_loss: 0.2180 - val_binary_accuracy: 0.9333
Epoch 6/10
44/44 [==============================] - 56s 1s/step - loss: 0.1941 -
binary_accuracy: 0.9391 - val_loss: 0.2410 - val_binary_accuracy: 0.9246
Epoch 7/10
44/44 [==============================] - 56s 1s/step - loss: 0.1421 -
binary_accuracy: 0.9507 - val_loss: 0.2142 - val_binary_accuracy: 0.9275
Epoch 8/10
44/44 [==============================] - 63s 1s/step - loss: 0.1144 -
binary_accuracy: 0.9572 - val_loss: 0.2634 - val_binary_accuracy: 0.9275
Epoch 9/10
44/44 [==============================] - 70s 2s/step - loss: 0.0846 -
binary_accuracy: 0.9710 - val_loss: 0.2323 - val_binary_accuracy: 0.9333
Epoch 10/10
44/44 [==============================] - 62s 1s/step - loss: 0.0741 -
binary_accuracy: 0.9768 - val_loss: 0.2440 - val_binary_accuracy: 0.9333
```

2.2.1 Evaluation of Model and plotting the results

[13]:

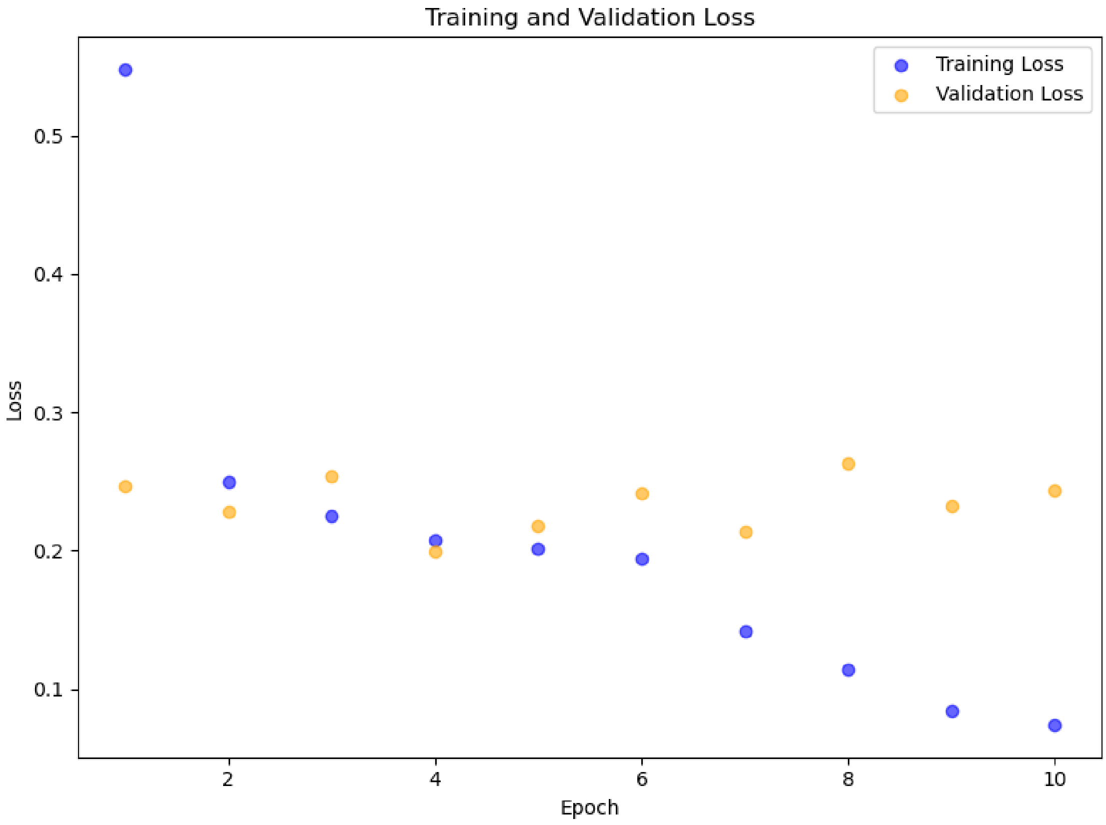
```
# Evaluate the model on the validation set
loss, accuracy = model.evaluate(X_val, y_val)
print('Validation loss:', loss)
print('Validation accuracy:', accuracy)
```

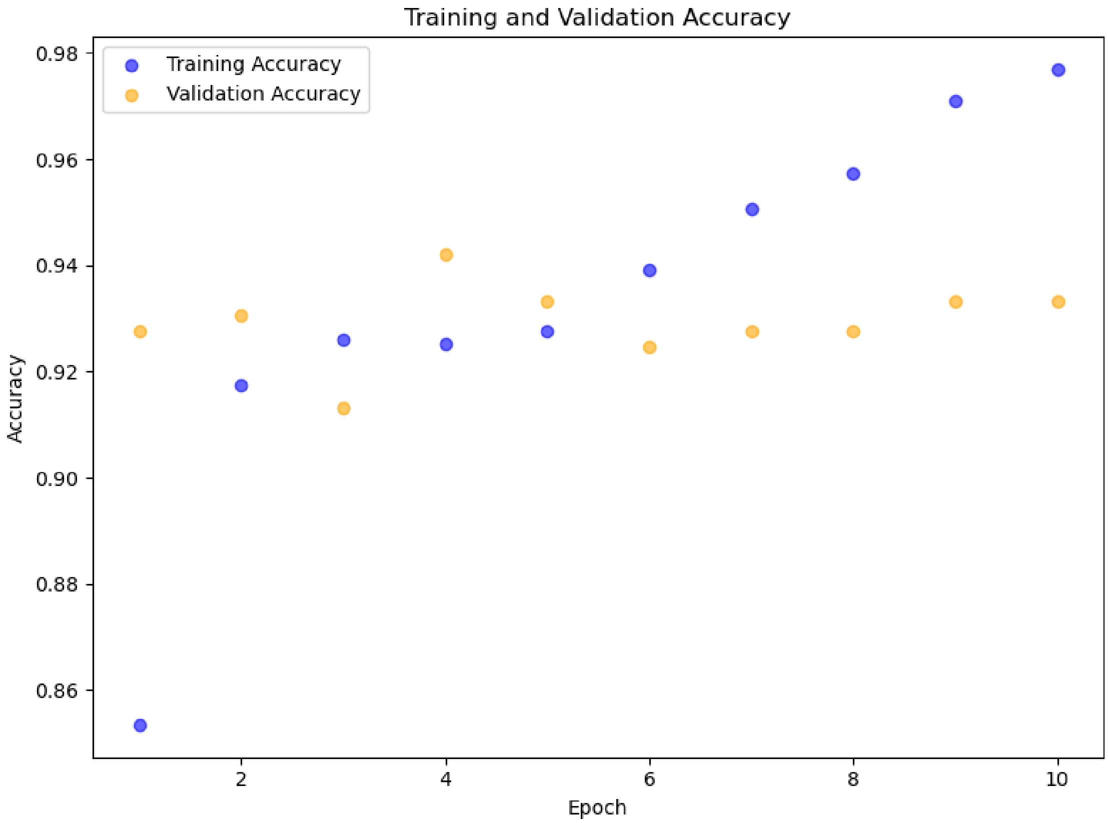
```
11/11 [==============================] - 3s 232ms/step - loss: 0.4153 -
```

```
binary_accuracy: 0.9391
Validation loss: 0.4152860641479492
Validation accuracy: 0.939130425453186
```

[21]:

```python
import matplotlib.pyplot as plt

epochs = range(1, len(history.history['loss']) + 1)

plt.figure(figsize=(8, 6))

plt.scatter(epochs, history.history['loss'], label='Training Loss', color='blue', alpha=0.6)
plt.scatter(epochs, history.history['val_loss'], label='Validation Loss', color='orange', alpha=0.6)

plt.title('Training and Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.tight_layout()
plt.show()

import matplotlib.pyplot as plt

epochs = range(1, len(history.history['binary_accuracy']) + 1)

plt.figure(figsize=(8, 6))

plt.scatter(epochs, history.history['binary_accuracy'], label='Training Accuracy', color='blue', alpha=0.6)
plt.scatter(epochs, history.history['val_binary_accuracy'], label='Validation Accuracy', color='orange', alpha=0.6)

plt.title('Training and Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.tight_layout()
plt.show()
```


Training and Validation Loss
● Training Loss
● Validation Loss
Loss
0.1
0.2
0.3
0.4
0.5
Epoch
2
4
6
8
10



2.2.2 Prediction on Unlabeled (test) data

[16]:

```python
# Make predictions on test images
predictions = model.predict(processed_data)

# Convert predictions to binary labels (0: benign, 1: malignant)
binary_predictions = (predictions > 0.5).astype(int)

for prediction in binary_predictions:
    if prediction == 1:
        print("benign")
    elif prediction == 0:
        print("malignant")
```

```
1/1 [==============================] - 0s 118ms/step
malignant
malignant
benign
malignant
malignant
malignant
```

```
benign
benign
benign
benign
malignant
malignant
benign
benign
```