Installing Packages

In [1]:
```
%pip install numpy
%pip install tensorflow==2.10.0
%pip install tensorflow-gpu==2.10.0
%pip install matplotlib
%pip install scikit-image
%pip install scikit-learn
%pip install joblib
```

```
Requirement already satisfied: numpy in c:\users\kelvi\anaconda3\envs\tf\lib\site-pa
ckages (1.24.3)
Note: you may need to restart the kernel to use updated packages.
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
```

```
Requirement already satisfied: tensorflow==2.10.0 in c:\users\kelvi\anaconda3\envs\t
f\lib\site-packages (2.10.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\kelvi\anaconda3\envs\tf\li
b\site-packages (from tensorflow==2.10.0) (2.1.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from tensorflow==2.10.0) (1.6.3)
Requirement already satisfied: flatbuffers>=2.0 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from tensorflow==2.10.0) (24.3.25)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from tensorflow==2.10.0) (0.4.0)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from tensorflow==2.10.0) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\kelvi\anaconda3\envs\tf\lib\s
ite-packages (from tensorflow==2.10.0) (3.11.0)
Requirement already satisfied: keras-preprocessing>=1.1.1 in c:\users\kelvi\anaconda
3\envs\tf\lib\site-packages (from tensorflow==2.10.0) (1.1.2)
Requirement already satisfied: libclang>=13.0.0 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from tensorflow==2.10.0) (18.1.1)
Requirement already satisfied: numpy>=1.20 in c:\users\kelvi\anaconda3\envs\tf\lib\s
ite-packages (from tensorflow==2.10.0) (1.24.3)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from tensorflow==2.10.0) (3.3.0)
Requirement already satisfied: packaging in c:\users\kelvi\appdata\roaming\python\py
thon310\site-packages (from tensorflow==2.10.0) (24.1)
Requirement already satisfied: protobuf<3.20,>=3.9.2 in c:\users\kelvi\anaconda3\env
s\tf\lib\site-packages (from tensorflow==2.10.0) (3.19.6)
Requirement already satisfied: setuptools in c:\users\kelvi\anaconda3\envs\tf\lib\si
te-packages (from tensorflow==2.10.0) (75.1.0)
Requirement already satisfied: six>=1.12.0 in c:\users\kelvi\appdata\roaming\python
\python310\site-packages (from tensorflow==2.10.0) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from tensorflow==2.10.0) (2.1.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\kelvi\appdata\ro
aming\python\python310\site-packages (from tensorflow==2.10.0) (4.12.2)
Requirement already satisfied: wrapt>=1.11.0 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from tensorflow==2.10.0) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in c:\users\kelv
i\anaconda3\envs\tf\lib\site-packages (from tensorflow==2.10.0) (0.31.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from tensorflow==2.10.0) (1.48.2)
Requirement already satisfied: tensorboard<2.11,>=2.10 in c:\users\kelvi\anaconda3\e
nvs\tf\lib\site-packages (from tensorflow==2.10.0) (2.10.1)
Requirement already satisfied: tensorflow-estimator<2.11,>=2.10.0 in c:\users\kelvi
\anaconda3\envs\tf\lib\site-packages (from tensorflow==2.10.0) (2.10.0)
Requirement already satisfied: keras<2.11,>=2.10.0 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from tensorflow==2.10.0) (2.10.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\kelvi\anaconda3\envs\t
f\lib\site-packages (from astunparse>=1.6.0->tensorflow==2.10.0) (0.44.0)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\kelvi\anaconda3\env
s\tf\lib\site-packages (from tensorboard<2.11,>=2.10->tensorflow==2.10.0) (2.29.0)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in c:\users\kelvi\an
aconda3\envs\tf\lib\site-packages (from tensorboard<2.11,>=2.10->tensorflow==2.10.0)
(0.4.4)
Requirement already satisfied: markdown>=2.6.8 in c:\users\kelvi\anaconda3\envs\tf\l
ib\site-packages (from tensorboard<2.11,>=2.10->tensorflow==2.10.0) (3.4.1)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\kelvi\anaconda3\envs
```

```
\tf\lib\site-packages (from tensorboard<2.11,>=2.10->tensorflow==2.10.0) (2.32.3)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in c:\users\kel
vi\anaconda3\envs\tf\lib\site-packages (from tensorboard<2.11,>=2.10->tensorflow==2.
10.0) (0.6.1)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in c:\users\kelvi\anaco
nda3\envs\tf\lib\site-packages (from tensorboard<2.11,>=2.10->tensorflow==2.10.0)
(1.8.1)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\kelvi\anaconda3\envs\tf\l
ib\site-packages (from tensorboard<2.11,>=2.10->tensorflow==2.10.0) (3.0.3)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\kelvi\anaconda3\en
vs\tf\lib\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.11,>=2.10->tensor
flow==2.10.0) (5.3.3)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\kelvi\anaconda3\env
s\tf\lib\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.11,>=2.10->tensorf
low==2.10.0) (0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.11,>=2.10->tensorflow==2.1
0.0) (4.7.2)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\kelvi\anaconda3
\envs\tf\lib\site-packages (from google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.11,
>=2.10->tensorflow==2.10.0) (2.0.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\kelvi\anaconda3
\envs\tf\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tenso
rflow==2.10.0) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tensorflow==2.10.
0) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\kelvi\anaconda3\envs\t
f\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tensorflow==
2.10.0) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\kelvi\anaconda3\envs\t
f\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tensorflow==
2.10.0) (2024.8.30)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from werkzeug>=1.0.1->tensorboard<2.11,>=2.10->tensorflow==2.10.
0) (2.1.3)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboar
d<2.11,>=2.10->tensorflow==2.10.0) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\kelvi\anaconda3\envs\tf\l
ib\site-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->t
ensorboard<2.11,>=2.10->tensorflow==2.10.0) (3.2.2)
Note: you may need to restart the kernel to use updated packages.
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
```

```
Requirement already satisfied: tensorflow-gpu==2.10.0 in c:\users\kelvi\anaconda3\en
vs\tf\lib\site-packages (2.10.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\kelvi\anaconda3\envs\tf\li
b\site-packages (from tensorflow-gpu==2.10.0) (2.1.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from tensorflow-gpu==2.10.0) (1.6.3)
Requirement already satisfied: flatbuffers>=2.0 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from tensorflow-gpu==2.10.0) (24.3.25)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from tensorflow-gpu==2.10.0) (0.4.0)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from tensorflow-gpu==2.10.0) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\kelvi\anaconda3\envs\tf\lib\s
ite-packages (from tensorflow-gpu==2.10.0) (3.11.0)
Requirement already satisfied: keras-preprocessing>=1.1.1 in c:\users\kelvi\anaconda
3\envs\tf\lib\site-packages (from tensorflow-gpu==2.10.0) (1.1.2)
Requirement already satisfied: libclang>=13.0.0 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from tensorflow-gpu==2.10.0) (18.1.1)
Requirement already satisfied: numpy>=1.20 in c:\users\kelvi\anaconda3\envs\tf\lib\s
ite-packages (from tensorflow-gpu==2.10.0) (1.24.3)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from tensorflow-gpu==2.10.0) (3.3.0)
Requirement already satisfied: packaging in c:\users\kelvi\appdata\roaming\python\py
thon310\site-packages (from tensorflow-gpu==2.10.0) (24.1)
Requirement already satisfied: protobuf<3.20,>=3.9.2 in c:\users\kelvi\anaconda3\env
s\tf\lib\site-packages (from tensorflow-gpu==2.10.0) (3.19.6)
Requirement already satisfied: setuptools in c:\users\kelvi\anaconda3\envs\tf\lib\si
te-packages (from tensorflow-gpu==2.10.0) (75.1.0)
Requirement already satisfied: six>=1.12.0 in c:\users\kelvi\appdata\roaming\python
\python310\site-packages (from tensorflow-gpu==2.10.0) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from tensorflow-gpu==2.10.0) (2.1.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\kelvi\appdata\ro
aming\python\python310\site-packages (from tensorflow-gpu==2.10.0) (4.12.2)
Requirement already satisfied: wrapt>=1.11.0 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from tensorflow-gpu==2.10.0) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in c:\users\kelv
i\anaconda3\envs\tf\lib\site-packages (from tensorflow-gpu==2.10.0) (0.31.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from tensorflow-gpu==2.10.0) (1.48.2)
Requirement already satisfied: tensorboard<2.11,>=2.10 in c:\users\kelvi\anaconda3\e
nvs\tf\lib\site-packages (from tensorflow-gpu==2.10.0) (2.10.1)
Requirement already satisfied: tensorflow-estimator<2.11,>=2.10.0 in c:\users\kelvi
\anaconda3\envs\tf\lib\site-packages (from tensorflow-gpu==2.10.0) (2.10.0)
Requirement already satisfied: keras<2.11,>=2.10.0 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from tensorflow-gpu==2.10.0) (2.10.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\kelvi\anaconda3\envs\t
f\lib\site-packages (from astunparse>=1.6.0->tensorflow-gpu==2.10.0) (0.44.0)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\kelvi\anaconda3\env
s\tf\lib\site-packages (from tensorboard<2.11,>=2.10->tensorflow-gpu==2.10.0) (2.29.
0)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in c:\users\kelvi\an
aconda3\envs\tf\lib\site-packages (from tensorboard<2.11,>=2.10->tensorflow-gpu==2.1
0.0) (0.4.4)
Requirement already satisfied: markdown>=2.6.8 in c:\users\kelvi\anaconda3\envs\tf\l
ib\site-packages (from tensorboard<2.11,>=2.10->tensorflow-gpu==2.10.0) (3.4.1)
```

```
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from tensorboard<2.11,>=2.10->tensorflow-gpu==2.10.0) (2.32.
3)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in c:\users\kel
vi\anaconda3\envs\tf\lib\site-packages (from tensorboard<2.11,>=2.10->tensorflow-gpu
==2.10.0) (0.6.1)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in c:\users\kelvi\anaco
nda3\envs\tf\lib\site-packages (from tensorboard<2.11,>=2.10->tensorflow-gpu==2.10.
0) (1.8.1)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\kelvi\anaconda3\envs\tf\l
ib\site-packages (from tensorboard<2.11,>=2.10->tensorflow-gpu==2.10.0) (3.0.3)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\kelvi\anaconda3\en
vs\tf\lib\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.11,>=2.10->tensor
flow-gpu==2.10.0) (5.3.3)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\kelvi\anaconda3\env
s\tf\lib\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.11,>=2.10->tensorf
low-gpu==2.10.0) (0.2.8)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from google-auth<3,>=1.6.3->tensorboard<2.11,>=2.10->tensorflow-gpu=
=2.10.0) (4.7.2)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\kelvi\anaconda3
\envs\tf\lib\site-packages (from google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.11,
>=2.10->tensorflow-gpu==2.10.0) (2.0.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\kelvi\anaconda3
\envs\tf\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tenso
rflow-gpu==2.10.0) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tensorflow-gpu==
2.10.0) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\kelvi\anaconda3\envs\t
f\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tensorflow-g
pu==2.10.0) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\kelvi\anaconda3\envs\t
f\lib\site-packages (from requests<3,>=2.21.0->tensorboard<2.11,>=2.10->tensorflow-g
pu==2.10.0) (2024.8.30)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from werkzeug>=1.0.1->tensorboard<2.11,>=2.10->tensorflow-gpu==
2.10.0) (2.1.3)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboar
d<2.11,>=2.10->tensorflow-gpu==2.10.0) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\kelvi\anaconda3\envs\tf\l
ib\site-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->t
ensorboard<2.11,>=2.10->tensorflow-gpu==2.10.0) (3.2.2)
Note: you may need to restart the kernel to use updated packages.
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
```

```
Requirement already satisfied: matplotlib in c:\users\kelvi\anaconda3\envs\tf\lib\si
te-packages (3.9.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from matplotlib) (1.3.0)
Requirement already satisfied: cycler>=0.10 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from matplotlib) (4.54.1)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from matplotlib) (1.4.7)
Requirement already satisfied: numpy>=1.23 in c:\users\kelvi\anaconda3\envs\tf\lib\s
ite-packages (from matplotlib) (1.24.3)
Requirement already satisfied: packaging>=20.0 in c:\users\kelvi\appdata\roaming\pyt
hon\python310\site-packages (from matplotlib) (24.1)
Requirement already satisfied: pillow>=8 in c:\users\kelvi\anaconda3\envs\tf\lib\sit
e-packages (from matplotlib) (11.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from matplotlib) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\kelvi\appdata\roamin
g\python\python310\site-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in c:\users\kelvi\appdata\roaming\python\pyt
hon310\site-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
Requirement already satisfied: scikit-image in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (0.24.0)
Requirement already satisfied: numpy>=1.23 in c:\users\kelvi\anaconda3\envs\tf\lib\s
ite-packages (from scikit-image) (1.24.3)
Requirement already satisfied: scipy>=1.9 in c:\users\kelvi\anaconda3\envs\tf\lib\si
te-packages (from scikit-image) (1.14.1)
Requirement already satisfied: networkx>=2.8 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from scikit-image) (3.4.1)
Requirement already satisfied: pillow>=9.1 in c:\users\kelvi\anaconda3\envs\tf\lib\s
ite-packages (from scikit-image) (11.0.0)
Requirement already satisfied: imageio>=2.33 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from scikit-image) (2.36.0)
Requirement already satisfied: tifffile>=2022.8.12 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from scikit-image) (2024.9.20)
Requirement already satisfied: packaging>=21 in c:\users\kelvi\appdata\roaming\pytho
n\python310\site-packages (from scikit-image) (24.1)
Requirement already satisfied: lazy-loader>=0.4 in c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages (from scikit-image) (0.4)
Note: you may need to restart the kernel to use updated packages.
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
```

```
Requirement already satisfied: scikit-learn in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (1.5.2)
Requirement already satisfied: numpy>=1.19.5 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from scikit-learn) (1.24.3)
Requirement already satisfied: scipy>=1.6.0 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from scikit-learn) (1.14.1)
Requirement already satisfied: joblib>=1.2.0 in c:\users\kelvi\anaconda3\envs\tf\lib
\site-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\users\kelvi\anaconda3\envs
\tf\lib\site-packages (from scikit-learn) (3.5.0)
Note: you may need to restart the kernel to use updated packages.
```

```
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
```

```
Requirement already satisfied: joblib in c:\users\kelvi\anaconda3\envs\tf\lib\site-p
ackages (1.4.2)Note: you may need to restart the kernel to use updated packages.
```

```
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
WARNING: Ignoring invalid distribution -ensorflow (c:\users\kelvi\anaconda3\envs\tf
\lib\site-packages)
```

In [2]:
```
# Common imports
import numpy as np;
import random;
import joblib;

# import other code files
from setup import load_images_paths, load_images, unpack, convertPredArray;
from performance import printConfMtx, getAccuracy, getPrecision, getRecall, getF1,
from MLP import MLP;
from CNN import CNN;

# to make this notebook's output stable across runs
np.random.seed(42);

# To plot pretty figures
%matplotlib inline
import matplotlib.pyplot as plt;
plt.rcParams['axes.labelsize'] = 14;
plt.rcParams['xtick.labelsize'] = 12;
plt.rcParams['ytick.labelsize'] = 12;

# import Global variables
import constants;
```

Setup

Load the Images

Load the Paths of the Images

In [3]:
```
image_paths = load_images_paths();
YOR_TEST_SAMPLES = image_paths["YOR_TEST_SAMPLES"];
YOR_TEST_LABELS = image_paths["YOR_TEST_LABELS"];
YOR_TRAIN_SAMPLES = image_paths["YOR_TRAIN_SAMPLES"];
YOR_TRAIN_LABELS = image_paths["YOR_TRAIN_LABELS"];

CAL_TEST_SAMPLES = image_paths["CAL_TEST_SAMPLES"];
CAL_TEST_LABELS = image_paths["CAL_TEST_LABELS"];
CAL_TRAIN_SAMPLES = image_paths["CAL_TRAIN_SAMPLES"];
CAL_TRAIN_LABELS = image_paths["CAL_TRAIN_LABELS"];

print("In YOR dataset...")
print(f"Number of Samples in test set: {len(YOR_TEST_SAMPLES)}");
print(f"Number of Labels in test set: {len(YOR_TEST_LABELS)}");
print(f"Number of Samples in training set: {len(YOR_TRAIN_SAMPLES)}");
print(f"Number of Labels in training set: {len(YOR_TRAIN_LABELS)}");

print();

print("In CAL dataset...")
print(f"Number of Samples in test set: {len(CAL_TEST_SAMPLES)}");
print(f"Number of Labels in test set: {len(CAL_TEST_LABELS)}");
print(f"Number of Samples in training set: {len(CAL_TRAIN_SAMPLES)}");
print(f"Number of Labels in training set: {len(CAL_TRAIN_LABELS)}");
```

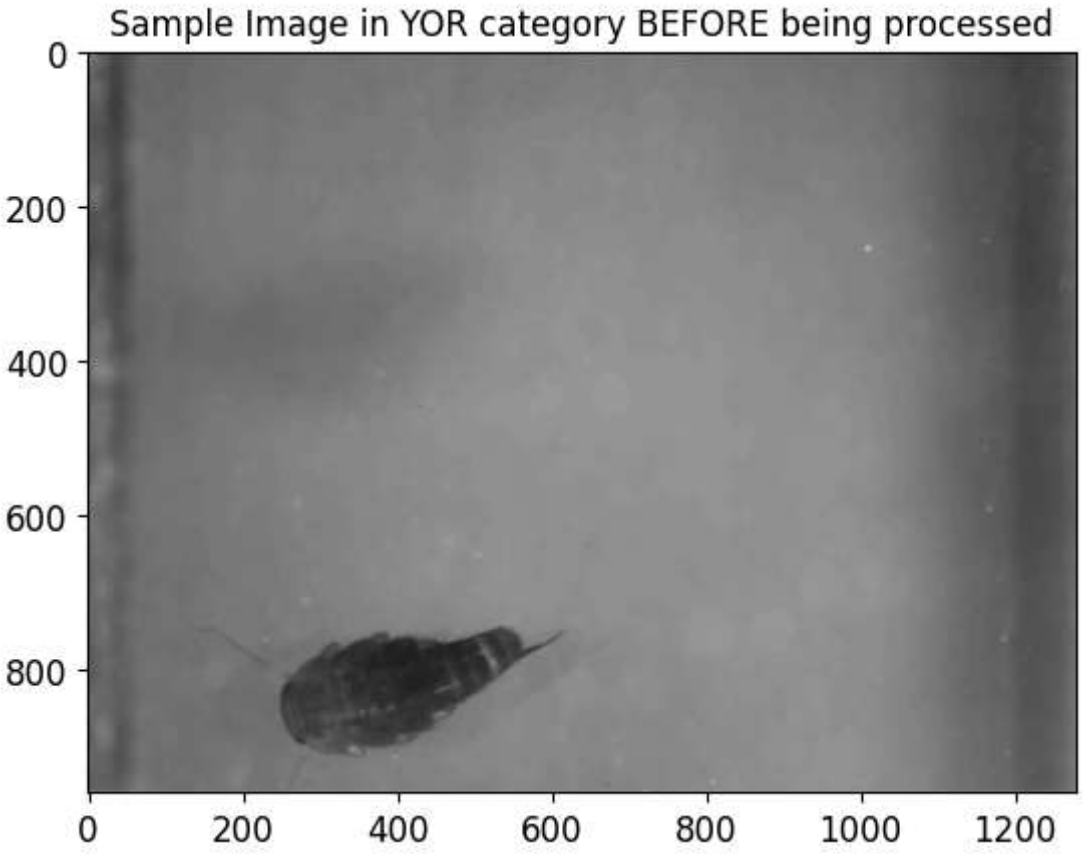
```
In YOR dataset...
Number of Samples in test set: 158
Number of Labels in test set: 158
Number of Samples in training set: 334
Number of Labels in training set: 334

In CAL dataset...
Number of Samples in test set: 237
Number of Labels in test set: 237
Number of Samples in training set: 253
Number of Labels in training set: 253
```

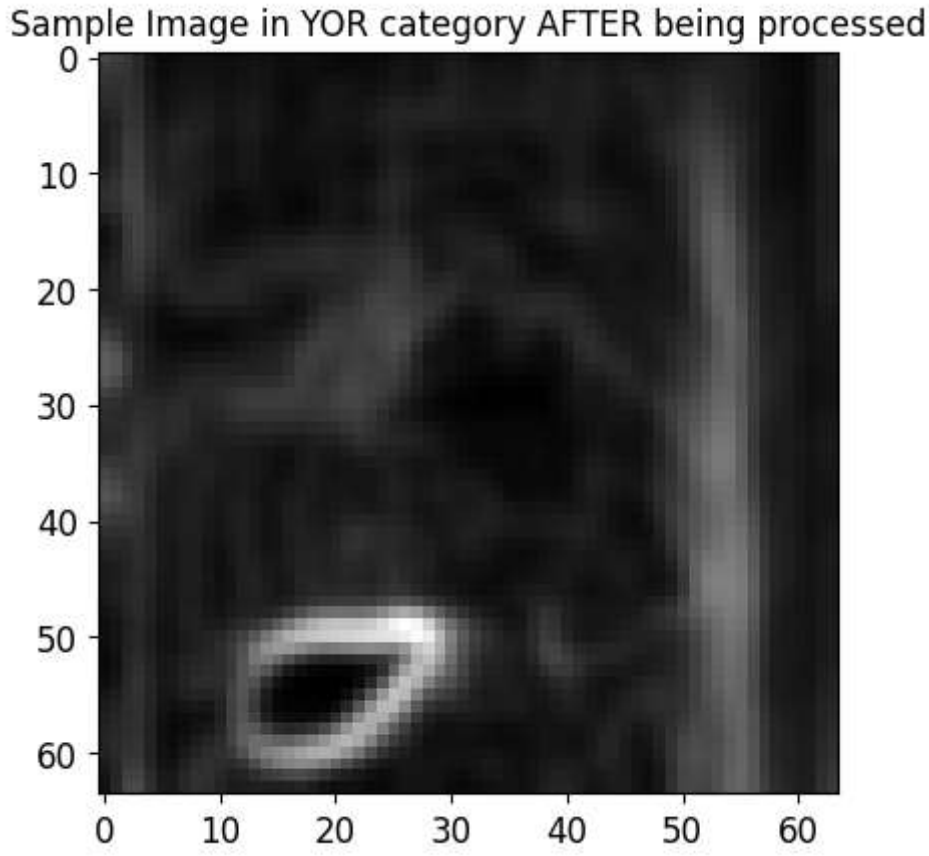
In [4]:
```
print("Inspecting the path in each set...");
print(YOR_TEST_SAMPLES[random.randint(0, len(YOR_TEST_SAMPLES)-1)]);
print(YOR_TRAIN_SAMPLES[random.randint(0, len(YOR_TRAIN_SAMPLES)-1)]);
print(CAL_TEST_SAMPLES[random.randint(0, len(CAL_TEST_SAMPLES)-1)]);
print(CAL_TRAIN_SAMPLES[random.randint(0, len(CAL_TRAIN_SAMPLES)-1)]);
```

```
Inspecting the path in each set...
.\stonefiles\yor\set0\Yor_20-specimen-i001-s124.jpg
.\stonefiles\yor\set1\Yor_110-specimen-i001-s125.jpg
.\stonefiles\cal\set0\Cal_32-specimen-i005-s125.jpg
.\stonefiles\cal\set2\Cal_84-specimen-i005-s125.jpg
```

Load the actual images (from the obtained paths) into numpy arrays

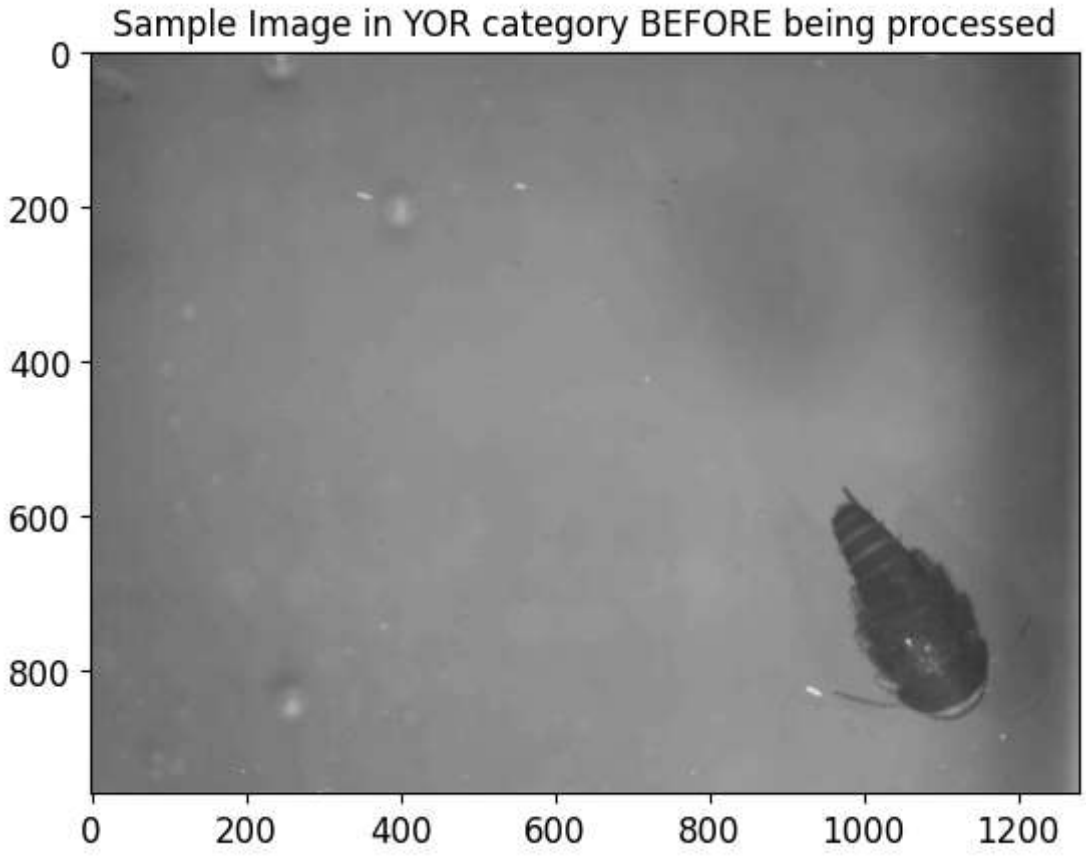
In [5]:
```
YOR_TEST_SAMPLES = load_images(YOR_TEST_SAMPLES, constants.YOR.upper());
print(f"Type of the test samples structure: {type(YOR_TEST_SAMPLES)}");
print(f"Shape of the test samples array: {YOR_TEST_SAMPLES.shape}");
print(f"Dimensionality of the test samples array: {YOR_TEST_SAMPLES.ndim}");
```

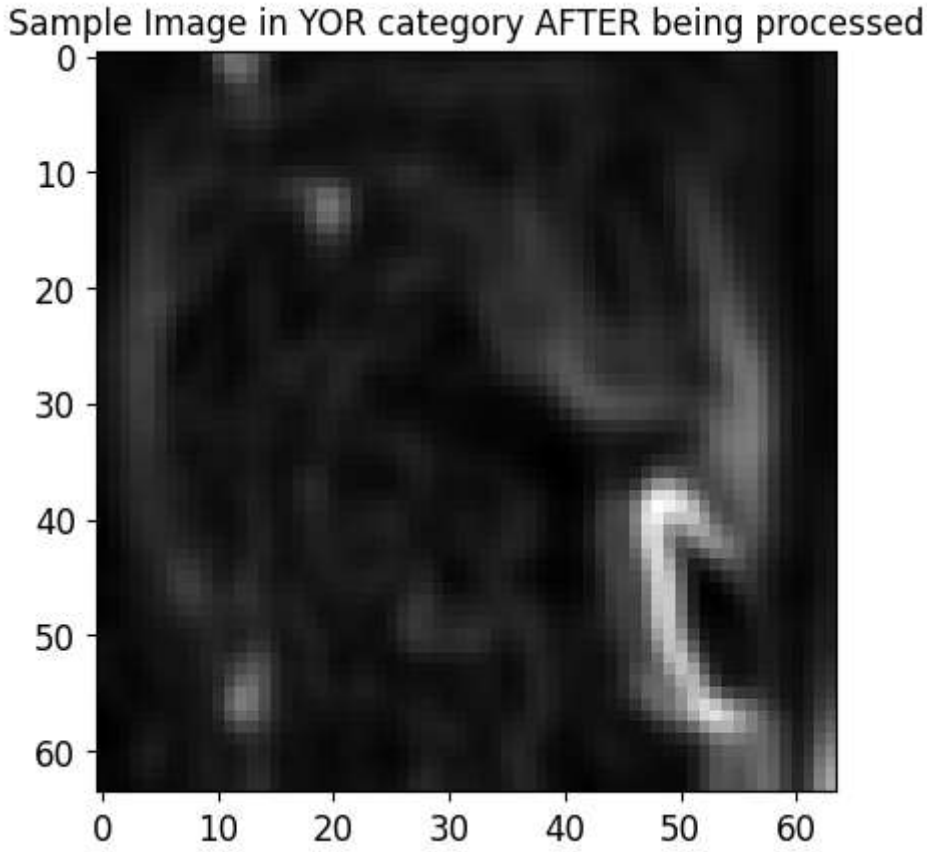





```
Type of the test samples structure: <class 'numpy.ndarray'>
Shape of the test samples array: (158, 64, 64)
Dimensionality of the test samples array: 3
```

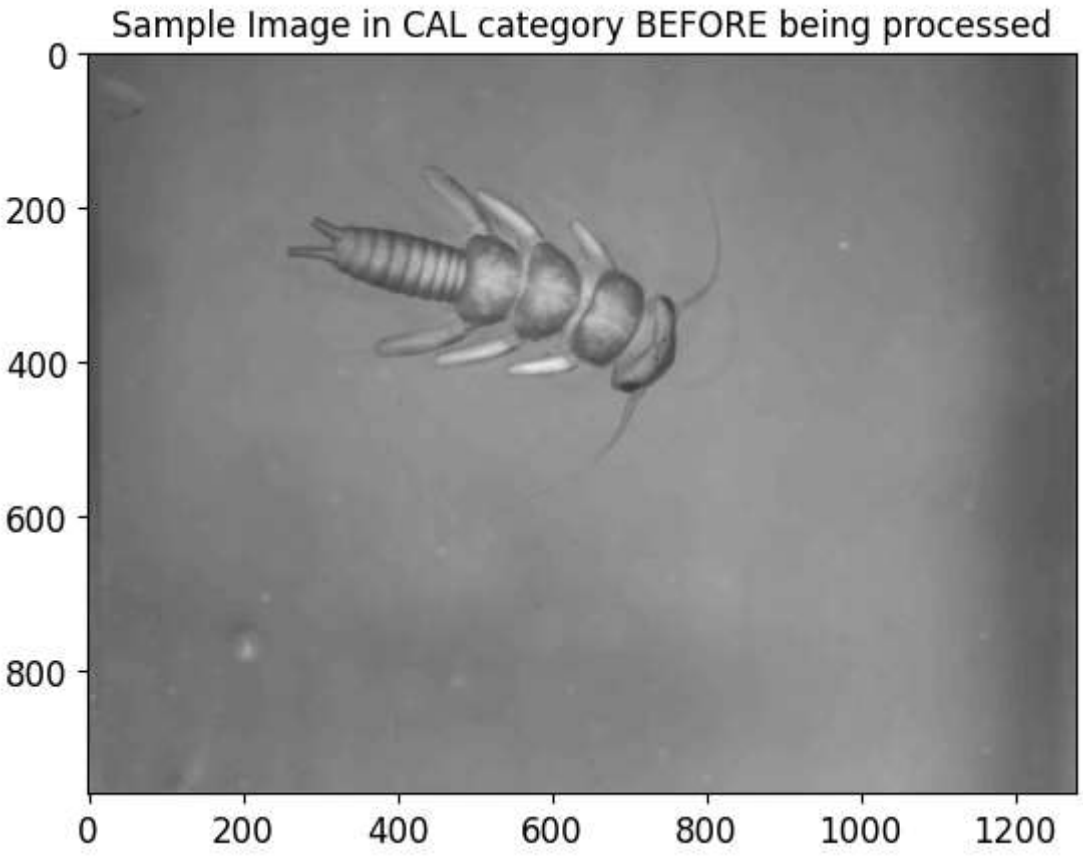
In [6]:
```
YOR_TRAIN_SAMPLES = load_images(YOR_TRAIN_SAMPLES, constants.YOR.upper());
print(f"Type of the training samples structure: {type(YOR_TRAIN_SAMPLES)}");
print(f"Shape of the training samples array: {YOR_TRAIN_SAMPLES.shape}");
print(f"Dimensionality of the training samples array: {YOR_TRAIN_SAMPLES.ndim}");
```

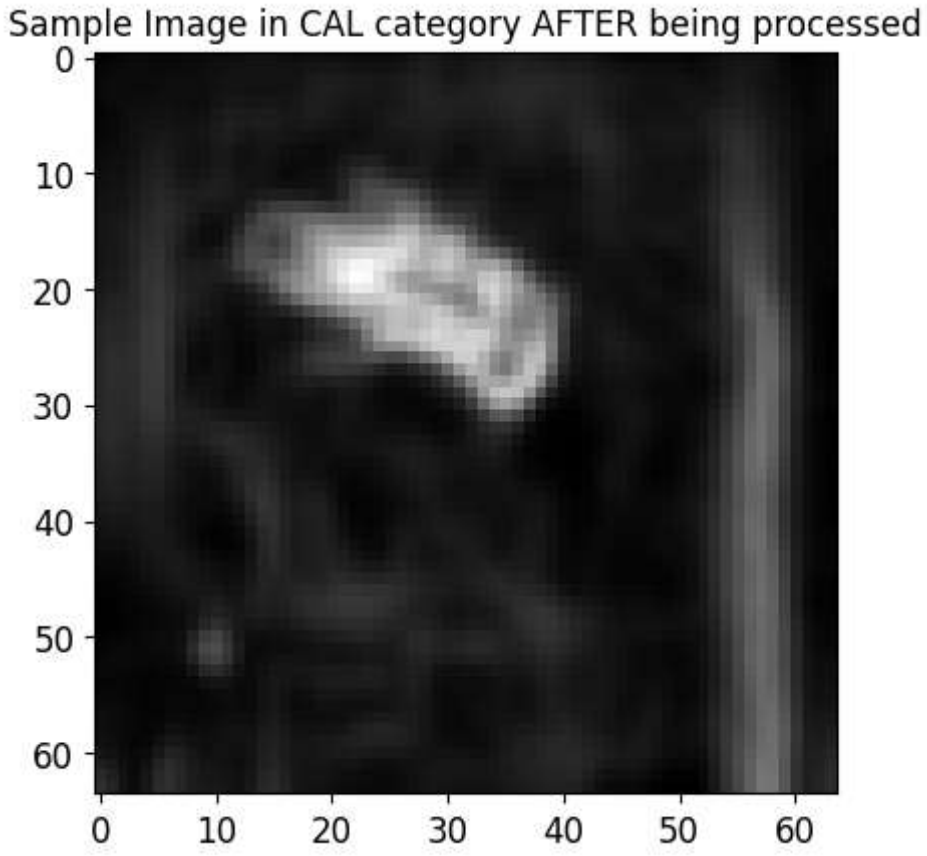





```
Type of the training samples structure: <class 'numpy.ndarray'>
Shape of the training samples array: (334, 64, 64)
Dimensionality of the training samples array: 3
```

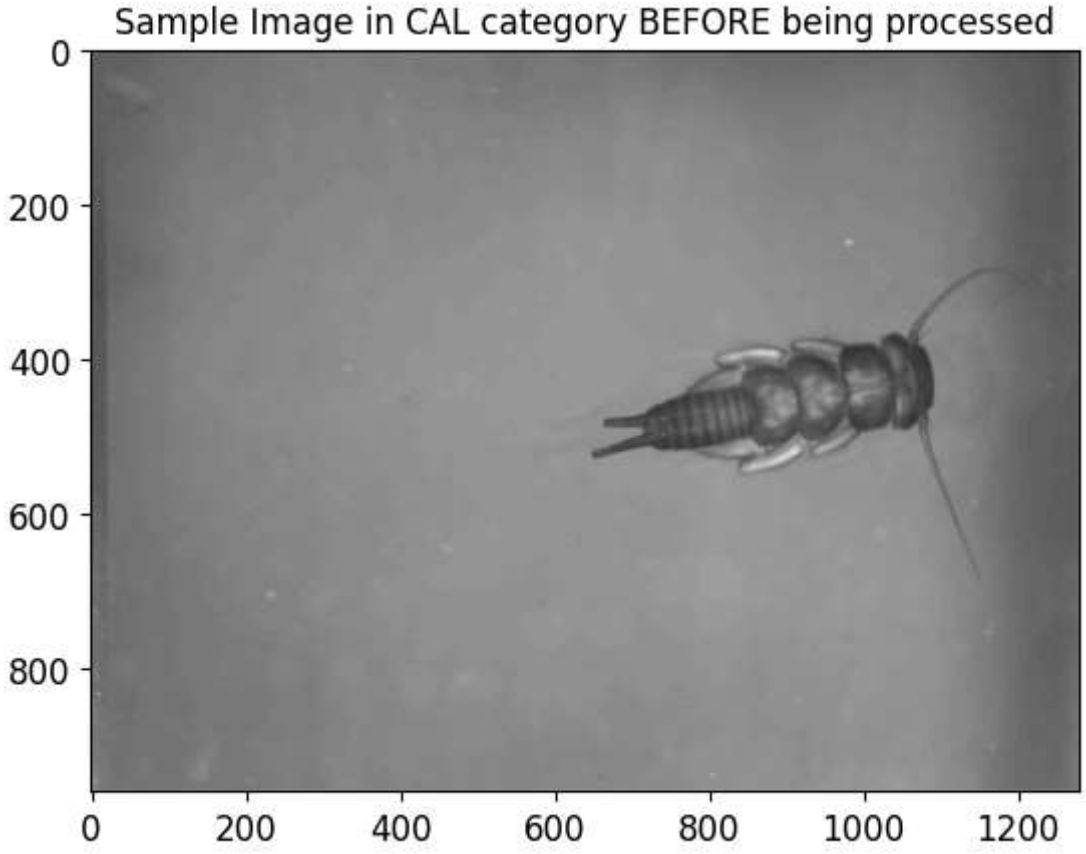
In [7]:
```
CAL_TEST_SAMPLES = load_images(CAL_TEST_SAMPLES, constants.CAL.upper());
print(f"Type of the test samples structure: {type(CAL_TEST_SAMPLES)}");
print(f"Shape of the test samples array: {CAL_TEST_SAMPLES.shape}");
print(f"Dimensionality of the test samples array: {CAL_TEST_SAMPLES.ndim}");
```

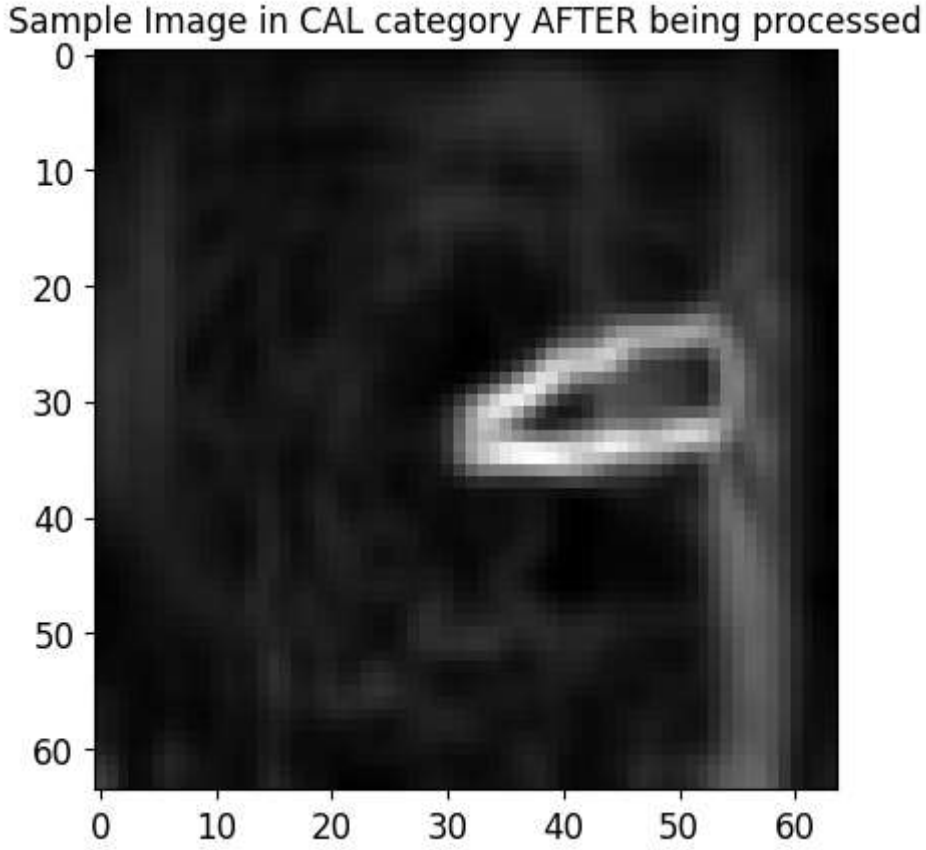





```
Type of the test samples structure: <class 'numpy.ndarray'>
Shape of the test samples array: (237, 64, 64)
Dimensionality of the test samples array: 3
```

In [8]:
```
CAL_TRAIN_SAMPLES = load_images(CAL_TRAIN_SAMPLES, constants.CAL.upper());
print(f"Type of the training samples structure: {type(CAL_TRAIN_SAMPLES)}");
print(f"Shape of the training samples array: {CAL_TRAIN_SAMPLES.shape}");
print(f"Dimensionality of the training samples array: {CAL_TRAIN_SAMPLES.ndim}");
```






```
Type of the training samples structure: <class 'numpy.ndarray'>
Shape of the training samples array: (253, 64, 64)
Dimensionality of the training samples array: 3
```

Put those into the appropriate training set and testing set for modelling

In [9]:
```
X_train = np.concatenate((YOR_TRAIN_SAMPLES, CAL_TRAIN_SAMPLES));
y_train = np.concatenate((YOR_TRAIN_LABELS, CAL_TRAIN_LABELS));
X_test = np.concatenate((YOR_TEST_SAMPLES, CAL_TEST_SAMPLES));
y_test = np.concatenate((YOR_TEST_LABELS, CAL_TEST_LABELS));
```

In [10]:
```
print(f"Shape of Training Set SamplesArray: {X_train.shape}");
print(f"Dimentionality of Training Set Samples Array: {X_train.ndim}");
print();
print(f"Shape of Training Set Labels Array: {y_train.shape}");
print(f"Dimentionality of Training Set Labels Array: {y_train.ndim}");

print();

print(f"Shape of Test Set SamplesArray: {X_test.shape}");
print(f"Dimentionality of Test Set Samples Array: {X_test.ndim}");
print();
print(f"Shape of Test Set Labels Array: {y_test.shape}");
print(f"Dimentionality of Test Set Labels Array: {y_test.ndim}");
```

```
Shape of Training Set SamplesArray: (587, 64, 64)
Dimentionality of Training Set Samples Array: 3

Shape of Training Set Labels Array: (587,)
Dimentionality of Training Set Labels Array: 1

Shape of Test Set SamplesArray: (395, 64, 64)
Dimentionality of Test Set Samples Array: 3

Shape of Test Set Labels Array: (395,)
Dimentionality of Test Set Labels Array: 1
```

In [11]:
```
# Save Images to files (for easier debugging)
joblib.dump(X_train, "X_train.pkl");
joblib.dump(X_test, "X_test.pkl");
joblib.dump(y_train, "y_train.pkl");
joblib.dump(y_test, "y_test.pkl");
```

In [12]:
```
# Test if the .pkl files store the image samples correctly
# X_train
unpackContent = unpack("X_train.pkl");
if (len(np.unique(unpackContent == X_train)) == 1 and np.unique(unpackContent == X_
    print("X_train in file is correct.");

# X_test
unpackContent = unpack("X_test.pkl");
if (len(np.unique(unpackContent == X_test)) == 1 and np.unique(unpackContent == X_t
    print("X_test in file is correct.");

# y_train
unpackContent = unpack("y_train.pkl");
if (len(np.unique(unpackContent == y_train)) == 1 and np.unique(unpackContent == y_
    print("y_train in file is correct.");
```

```
# y_test
unpackContent = unpack("y_test.pkl");
if (len(np.unique(unpackContent == y_test)) == 1 and np.unique(unpackContent == y_t
    print("y_test in file is correct.");
```

```
X_train in file is correct.
X_test in file is correct.
y_train in file is correct.
y_test in file is correct.
y_train in file is correct.
y_test in file is correct.
```

In [13]:
```
X_train = unpack("X_train.pkl");
X_test = unpack("X_test.pkl");
y_train = unpack("y_train.pkl");
y_test = unpack("y_test.pkl");
```

Multi-Layer Perceptron

In [14]:
```
# Load the Model
mlpModel = MLP();
mlpModel.model
```

```
WARNING:tensorflow:Compiled the loaded model, but the compiled metrics have yet to b
e built. `model.compile_metrics` will be empty until you train or evaluate the mode
l.
INFO:tensorflow:Assets written to: ram://f17d7c8f-7b20-4dfb-819e-3a813ff73989/assets
INFO:tensorflow:Assets written to: ram://3a6f0874-3078-4766-a987-05f26f1d2c89/assets
```

Out[14]:
```
<keras.engine.sequential.Sequential at 0x17da53c21d0>
```

In [15]:
```
# Normalize data
X_train, y_train, X_test, y_test = mlpModel.normalize(
    X_train=X_train,
    y_train=y_train,
    X_test=X_test,
    y_test=y_test
)
```

In [16]:
```
# summarize the model
mlpModel.model.summary();
```

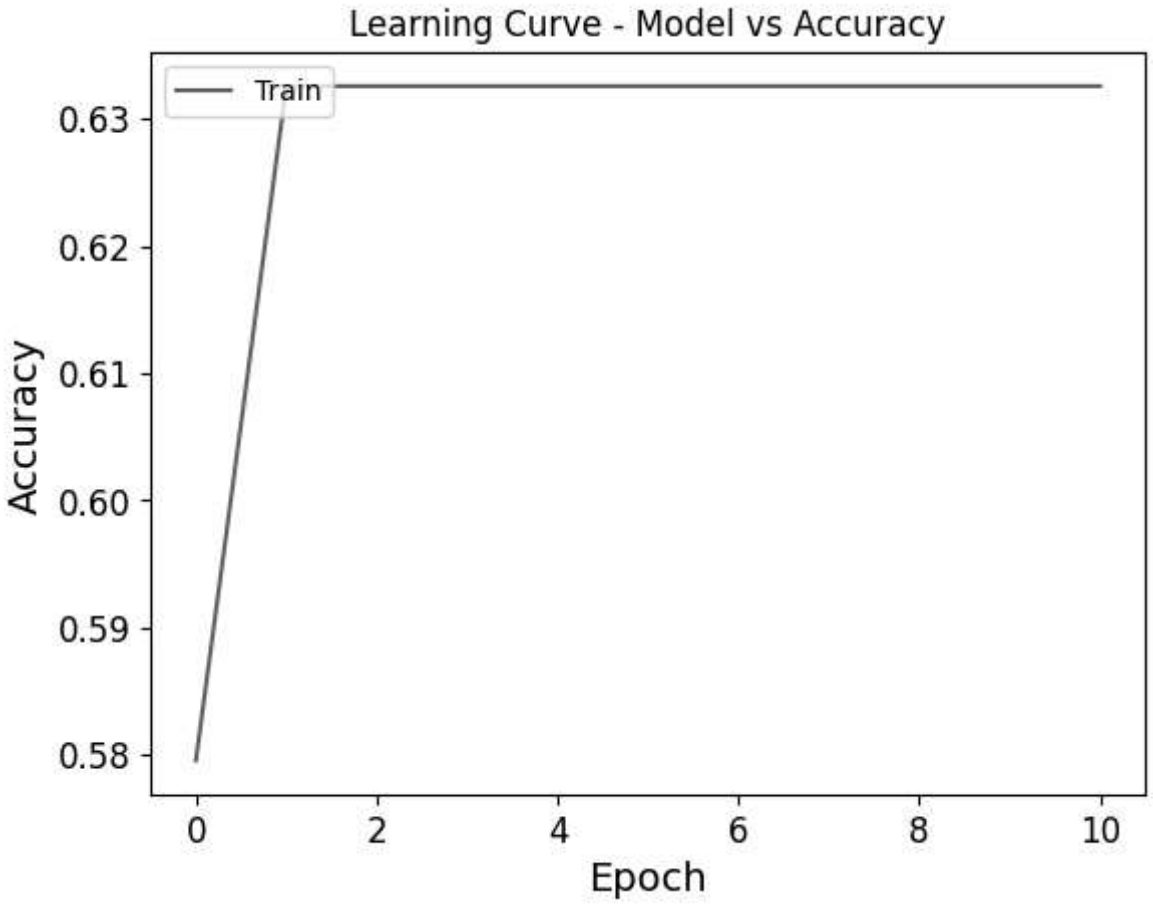
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 4096)              0

 dense (Dense)               (None, 64)                262208
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 4096)              0

 dense (Dense)               (None, 64)                262208

 dense_1 (Dense)             (None, 1)                 65

=================================================================
Total params: 262,273
Trainable params: 262,273
Non-trainable params: 0
_________________________________________________________________
```

In [17]:
```
# Train the Model
history = mlpModel.train(
    X_train=X_train,
    y_train=y_train,
    X_test=X_test,
    y_test=y_test
);
mlpModel.model
```

```
-----started training MLP model-----
INFO:tensorflow:Assets written to: ram://20c5fbb2-c36d-4b83-bfda-4c86c868e9e6/assets
13/13 [==============================] - 0s 2ms/step - loss: 0.6983 - acc: 0.4000
Test accuracy: 40.0%
-----The MLP model is completedly trained-----
Training Time Elapsed: 0:00:02.096241
```

Out[17]:
```
<keras.engine.sequential.Sequential at 0x17da53c21d0>
```

In [18]:
```
# Plot the Learning Curve
MLP.plotLearningCurve(history);
```



Prediction based on Training Set

In [19]:
```
y_pred = mlpModel.predict(X_test=X_train);
print(f"Number of predictions made: {len(y_pred)}");
print(f"Shape of predictions array: {y_pred.shape}");
print(f"Dimentionality of predictions array: {y_pred.ndim}");
```

```
-----Started Predicting test samples-----
19/19 [==============================] - 0s 1ms/step
-----Prediction completed-----
Prediction Time Elapsed: 0:00:00.161907
Number of predictions made: 587
Shape of predictions array: (587, 1)
Dimentionality of predictions array: 2
```

Since the predictions array is in 2-dimension, we have to inspect an element in it and perform some conversions to the array.

In [20]:
```
y_pred[random.randint(0, len(y_pred))]
```

Out[20]:
```
array([0.5102707], dtype=float32)
```

We now understood that an element is basically a number (between 0 and 1) being wrapped in a list. Therefore, we now perform the conversion to `y_pred`.

In [21]:
```
y_pred = convertPredArray(y_pred);
# Convert predictions to class labels (0 or 1) based on a threshold of 0.5
y_pred = (y_pred > 0.5).astype(int);
print(f"Number of predictions made: {len(y_pred)}");
print(f"Shape of predictions array: {y_pred.shape}");
print(f"Dimentionality of predictions array: {y_pred.ndim}");
print(f"An element in the converted predictions array now: {y_pred[random.randint(0
```

```
Number of predictions made: 587
Shape of predictions array: (587,)
Dimentionality of predictions array: 1
An element in the converted predictions array now: 1
```

In [22]:
```
# Recap
print(f"Number of actual samples in the test set: {len(X_train)}");
print(f"Number of actual labels in the test set: {len(y_train)}");
print(f"Shape of predictions array: {y_train.shape}");
```

```
Number of actual samples in the test set: 587
Number of actual labels in the test set: 587
Shape of predictions array: (587,)
```

We know that all samples have been successfully predicted from the training set. Now we display the confusion matrix regarding the prediction just made based on the Training Set.

In [23]:
```
printConfMtx(
    y_pred=y_pred,
    y_test=y_train
)
```

Out[23]:

Actual	Predicted Yor	Cal
Yor	334	0
Cal	253	0

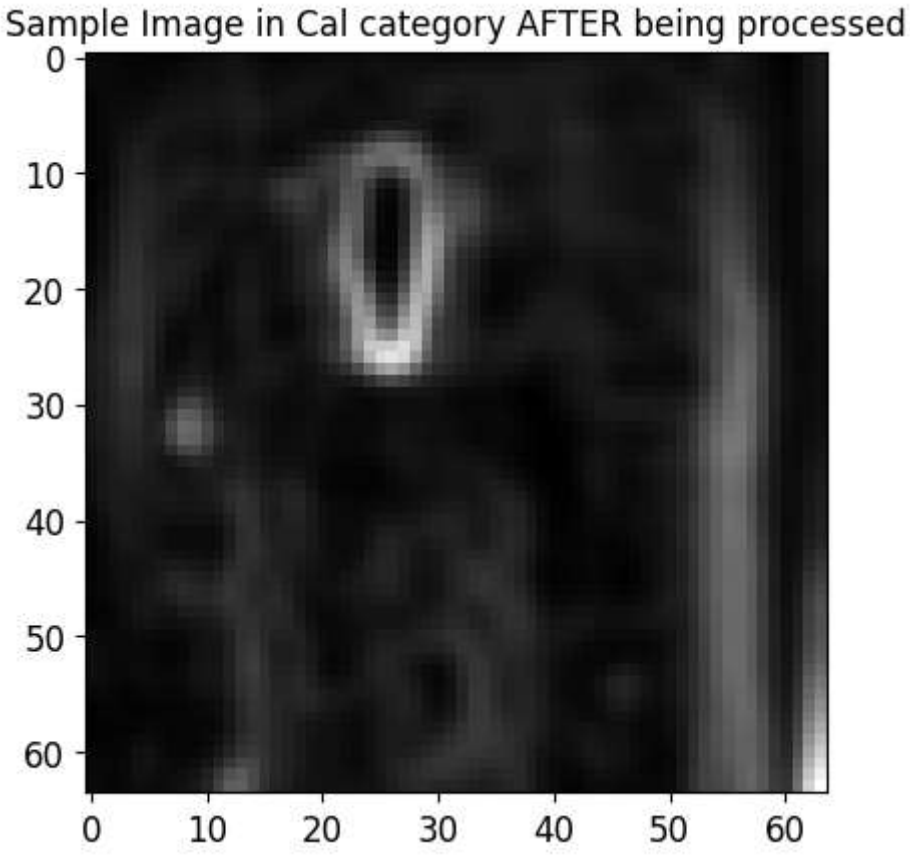
In [24]:
```
# Evaluate Metrics
# first item is for the metric score in training set; second item is for that in tr
accuracies_mlp, precisions_mlp, recalls_mlp, f1s_mlp = [], [], [], [];
accuracies_mlp.append(getAccuracy(
    y_pred=y_pred,
    y_test=y_train
));
precisions_mlp.append(getPrecision(
    y_pred=y_pred,
    y_test=y_train
));
recalls_mlp.append(getRecall(
    y_pred=y_pred,
    y_test=y_train
));
f1s_mlp.append(getF1(
    y_pred=y_pred,
```

```
    y_test=y_train
));
print(f"Accuracy of this MLP model based on Training Set: {accuracies_mlp[0]} ");
print(f"Precision of this MLP model based on Training Set: {precisions_mlp[0]} ");
print(f"Recall of this MLP model based on Training Set: {recalls_mlp[0]} ");
print(f"F1 Score of this MLP model based on Training Set: {f1s_mlp[0]} ");
```

```
Accuracy of this MLP model based on Training Set: 0.5689948892674617
Precision of this MLP model based on Training Set: 0.5689948892674617
Recall of this MLP model based on Training Set: 1.0
F1 Score of this MLP model based on Training Set: 0.7252985884907709
```

Visualizing a Misclassified Image In the Training Set

In [25]:
```
visualizeMisclassified(
    X_test=X_train,
    y_test=y_train,
    y_pred=y_pred
);
```




```
Number of misclassified images for this model: 253 out of total 587 images in the se
t.
```

Prediction based on Test Set

In [26]:
```
y_pred = mlpModel.predict(X_test=X_test);
print(f"Number of predictions made: {len(y_pred)}");
print(f"Shape of predictions array: {y_pred.shape}");
print(f"Dimentionality of predictions array: {y_pred.ndim}");
```

```
-----Started Predicting test samples-----
13/13 [==============================] - 0s 1ms/step
-----Prediction completed-----
Prediction Time Elapsed: 0:00:00.079966
Number of predictions made: 395
Shape of predictions array: (395, 1)
Dimentionality of predictions array: 2
```

Since the predictions array is in 2-dimension, we have to inspect an element in it and perform some conversions to the array.

In [27]:
```
y_pred[random.randint(0, len(y_pred))]
```

Out[27]:
```
array([0.5115164], dtype=float32)
```

We now understood that an element is basically a number (between 0 and 1) being wrapped in a list. Therefore, we now perform the conversion to `y_pred`.

In [28]:
```
y_pred = convertPredArray(y_pred);
# Convert predictions to class labels (0 or 1) based on a threshold of 0.5
y_pred = (y_pred > 0.5).astype(int);
print(f"Number of predictions made: {len(y_pred)}");
print(f"Shape of predictions array: {y_pred.shape}");
print(f"Dimentionality of predictions array: {y_pred.ndim}");
print(f"An element in the converted predictions array now: {y_pred[random.randint(0
```

```
Number of predictions made: 395
Shape of predictions array: (395,)
Dimentionality of predictions array: 1
An element in the converted predictions array now: 1
```

In [29]:
```
# Recap
print(f"Number of actual samples in the test set: {len(X_test)}");
print(f"Number of actual labels in the test set: {len(y_test)}");
print(f"Shape of predictions array: {y_test.shape}");
```

```
Number of actual samples in the test set: 395
Number of actual labels in the test set: 395
Shape of predictions array: (395,)
```

We know that all samples have been successfully predicted from the training set. Now we display the confusion matrix regarding the prediction just made based on the Test Set.

In [30]:
```
printConfMtx(
    y_pred=y_pred,
    y_test=y_test
)
```

Out[30]:

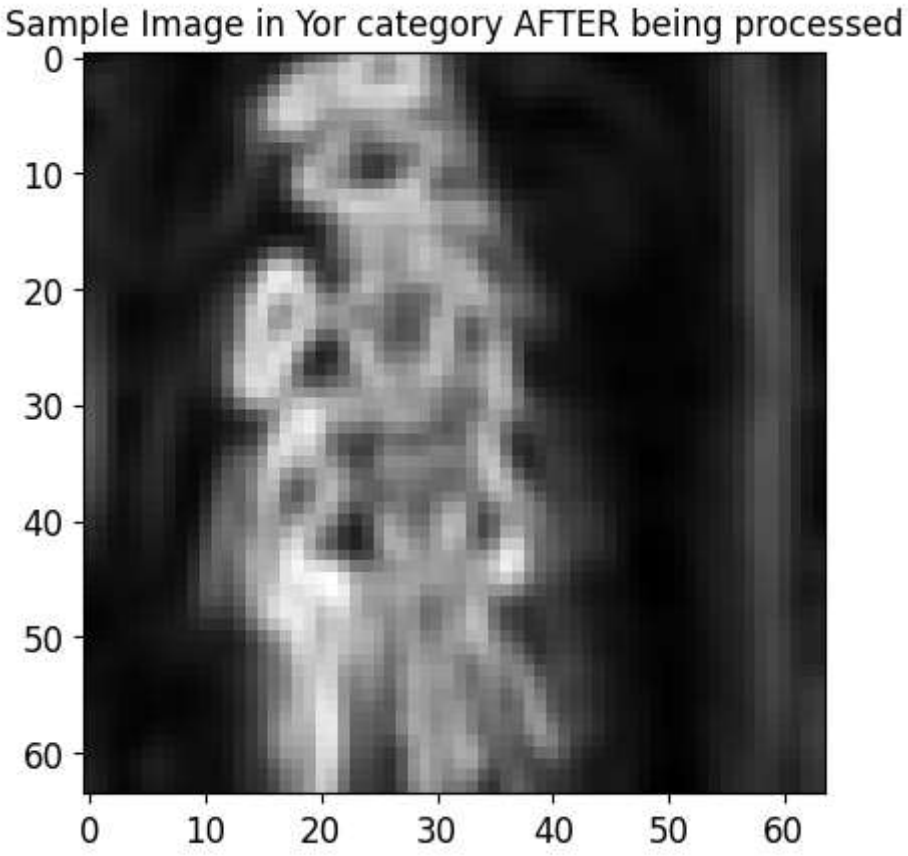
Actual	Predicted Yor	Predicted Cal
Yor	158	0
Cal	237	0

In [31]:
```
# Evaluate Metrics
# first item is for the metric score in training set; second item is for that in te
accuracies_mlp.append(getAccuracy(
    y_pred=y_pred,
    y_test=y_test
));
precisions_mlp.append(getPrecision(
    y_pred=y_pred,
    y_test=y_test
));
recalls_mlp.append(getRecall(
    y_pred=y_pred,
    y_test=y_test
));
f1s_mlp.append(getF1(
    y_pred=y_pred,
    y_test=y_test
));
print(f"Accuracy of this MLP model based on Test Set: {accuracies_mlp[-1]} ");
print(f"Precision of this MLP model based on Test Set: {precisions_mlp[-1]} ");
print(f"Recall of this MLP model based on Test Set: {recalls_mlp[-1]} ");
print(f"F1 Score of this MLP model based on Test Set: {f1s_mlp[-1]} ");
```

```
Accuracy of this MLP model based on Test Set: 0.4
Precision of this MLP model based on Test Set: 0.4
Recall of this MLP model based on Test Set: 1.0
F1 Score of this MLP model based on Test Set: 0.5714285714285714
```

Visualizing a Misclassified Image in the Test Set

In [32]:
```
visualizeMisclassified(
    X_test=X_test,
    y_test=y_test,
    y_pred=y_pred
);
```




```
Number of misclassified images for this model: 237 out of total 395 images in the se
t.
```

Convolutional Neural Network

In [33]:
```
# Build the model
cnnModel = CNN();
cnnModel.model;
```

```
WARNING:tensorflow:Compiled the loaded model, but the compiled metrics have yet to b
e built. `model.compile_metrics` will be empty until you train or evaluate the mode
l.
WARNING:absl:Found untraced functions such as _jit_compiled_convolution_op, _jit_com
piled_convolution_op, _jit_compiled_convolution_op while saving (showing 3 of 3). Th
ese functions will not be directly callable after loading.
INFO:tensorflow:Assets written to: ram://aef9d7ec-35a1-4c9d-afc2-1737683306ad/assets
INFO:tensorflow:Assets written to: ram://aef9d7ec-35a1-4c9d-afc2-1737683306ad/assets
WARNING:absl:Found untraced functions such as _jit_compiled_convolution_op, _jit_com
piled_convolution_op, _jit_compiled_convolution_op while saving (showing 3 of 3). Th
ese functions will not be directly callable after loading.
INFO:tensorflow:Assets written to: ram://6d201fcd-6104-4260-95ef-29b5f40f6cbe/assets
INFO:tensorflow:Assets written to: ram://6d201fcd-6104-4260-95ef-29b5f40f6cbe/assets
```

In [34]:
```
# summarize the model
cnnModel.model.summary();
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_1 (Conv2D)           (None, 62, 62, 64)        640

 max_pooling2d_1 (MaxPooling  (None, 31, 31, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 29, 29, 64)        36928

 max_pooling2d_2 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 conv2d_3 (Conv2D)           (None, 12, 12, 64)        36928

 max_pooling2d_3 (MaxPooling  (None, 6, 6, 64)         0

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_1 (Conv2D)           (None, 62, 62, 64)        640

 max_pooling2d_1 (MaxPooling  (None, 31, 31, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 29, 29, 64)        36928

 max_pooling2d_2 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 conv2d_3 (Conv2D)           (None, 12, 12, 64)        36928

 max_pooling2d_3 (MaxPooling  (None, 6, 6, 64)         0
 2D)

 flatten_1 (Flatten)         (None, 2304)              0

 dense_2 (Dense)             (None, 1)                 2305

=================================================================
Total params: 76,801
Trainable params: 76,801
Non-trainable params: 0
_________________________________________________________________
```

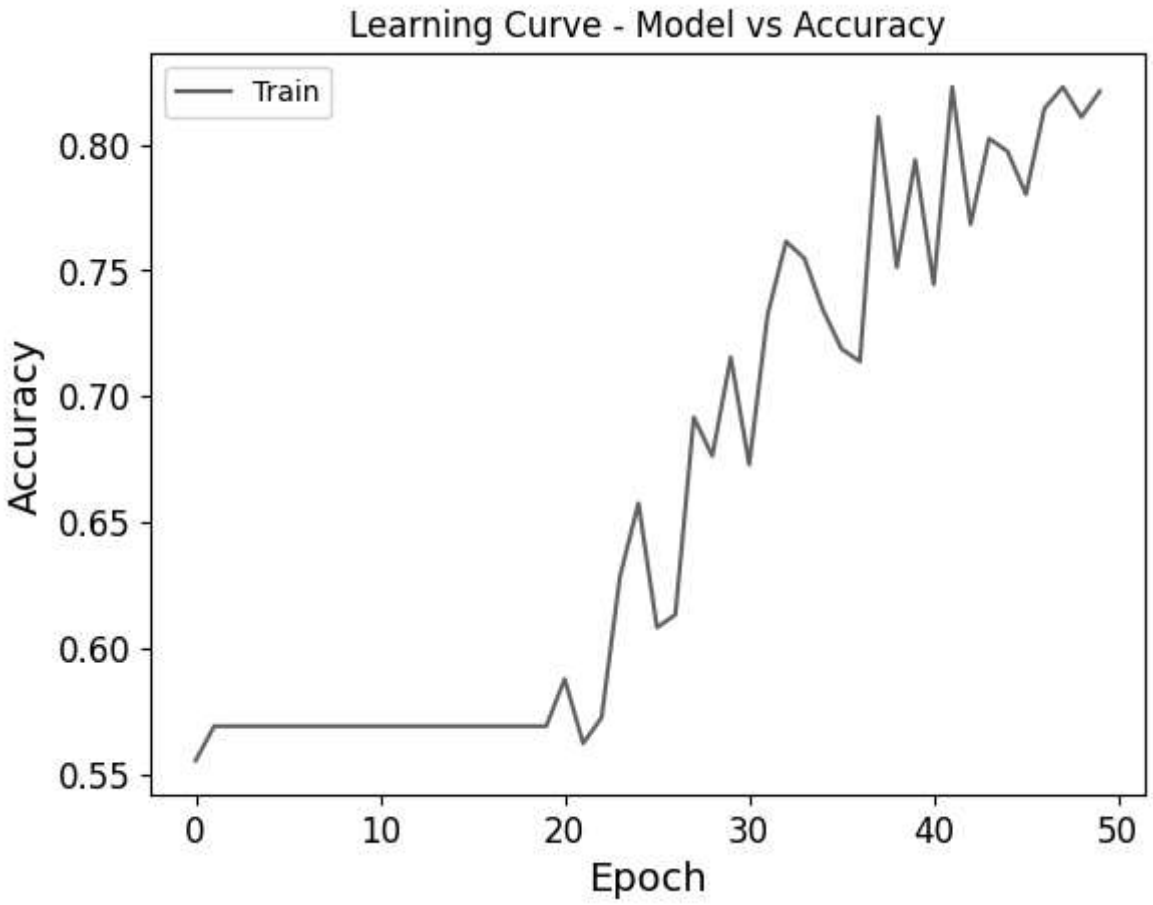
Note: Since the data has already been normalized before training our first model (the MLP model), we need not to normalize again here.

In [35]:
```
# Train the model
history = cnnModel.train(
    X_train=X_train,
    y_train=y_train,
    X_test=X_test,
    y_test=y_test
);
```

```
-----started training CNN model-----
Epoch 1/50
8/8 [==============================] - 3s 90ms/step - loss: 0.6995 - acc: 0.5554
Epoch 2/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6862 - acc: 0.5690
Epoch 3/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6850 - acc: 0.5690
Epoch 4/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6843 - acc: 0.5690
Epoch 5/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6853 - acc: 0.5690
Epoch 6/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6877 - acc: 0.5690
Epoch 7/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6853 - acc: 0.5690
Epoch 8/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6840 - acc: 0.5690
Epoch 9/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6837 - acc: 0.5690
Epoch 10/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6856 - acc: 0.5690
Epoch 11/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6842 - acc: 0.5690
Epoch 12/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6876 - acc: 0.5690
Epoch 13/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6860 - acc: 0.5690
Epoch 14/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6837 - acc: 0.5690
Epoch 15/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6846 - acc: 0.5690
Epoch 16/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6859 - acc: 0.5690
Epoch 17/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6834 - acc: 0.5690
Epoch 18/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6846 - acc: 0.5690
Epoch 19/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6810 - acc: 0.5690
Epoch 20/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6772 - acc: 0.5690
Epoch 21/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6794 - acc: 0.5877
Epoch 22/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6781 - acc: 0.5622
Epoch 23/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6679 - acc: 0.5724
Epoch 24/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6686 - acc: 0.6286
Epoch 25/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6478 - acc: 0.6576
Epoch 26/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6599 - acc: 0.6082
Epoch 27/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6471 - acc: 0.6133
Epoch 28/50
```

```
8/8 [==============================] - 0s 23ms/step - loss: 0.6228 - acc: 0.6917
Epoch 29/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6177 - acc: 0.6763
Epoch 30/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6025 - acc: 0.7155
Epoch 31/50
8/8 [==============================] - 0s 23ms/step - loss: 0.6196 - acc: 0.6729
Epoch 32/50
8/8 [==============================] - 0s 23ms/step - loss: 0.5765 - acc: 0.7325
Epoch 33/50
8/8 [==============================] - 0s 23ms/step - loss: 0.5622 - acc: 0.7615
Epoch 34/50
8/8 [==============================] - 0s 23ms/step - loss: 0.5557 - acc: 0.7547
Epoch 35/50
8/8 [==============================] - 0s 23ms/step - loss: 0.5546 - acc: 0.7342
Epoch 36/50
8/8 [==============================] - 0s 23ms/step - loss: 0.5494 - acc: 0.7189
Epoch 37/50
8/8 [==============================] - 0s 23ms/step - loss: 0.5639 - acc: 0.7138
Epoch 38/50
8/8 [==============================] - 0s 23ms/step - loss: 0.4932 - acc: 0.8109
Epoch 39/50
8/8 [==============================] - 0s 23ms/step - loss: 0.5303 - acc: 0.7513
Epoch 40/50
8/8 [==============================] - 0s 23ms/step - loss: 0.4914 - acc: 0.7939
Epoch 41/50
8/8 [==============================] - 0s 23ms/step - loss: 0.5294 - acc: 0.7445
Epoch 42/50
8/8 [==============================] - 0s 24ms/step - loss: 0.4603 - acc: 0.8228
Epoch 43/50
8/8 [==============================] - 0s 23ms/step - loss: 0.5073 - acc: 0.7683
Epoch 44/50
8/8 [==============================] - 0s 23ms/step - loss: 0.4849 - acc: 0.8024
Epoch 45/50
8/8 [==============================] - 0s 23ms/step - loss: 0.4609 - acc: 0.7973
Epoch 46/50
8/8 [==============================] - 0s 23ms/step - loss: 0.4757 - acc: 0.7802
Epoch 47/50
8/8 [==============================] - 0s 23ms/step - loss: 0.4433 - acc: 0.8143
Epoch 48/50
8/8 [==============================] - 0s 23ms/step - loss: 0.4373 - acc: 0.8228
Epoch 49/50
8/8 [==============================] - 0s 23ms/step - loss: 0.4399 - acc: 0.8109
Epoch 50/50
8/8 [==============================] - 0s 23ms/step - loss: 0.4202 - acc: 0.8211
WARNING:absl:Found untraced functions such as _jit_compiled_convolution_op, _jit_com
piled_convolution_op, _jit_compiled_convolution_op while saving (showing 3 of 3). Th
ese functions will not be directly callable after loading.
INFO:tensorflow:Assets written to: ram://a049edab-31ad-423c-9b5d-a81018a51402/assets
INFO:tensorflow:Assets written to: ram://a049edab-31ad-423c-9b5d-a81018a51402/assets
13/13 [==============================] - 0s 12ms/step - loss: 0.3198 - acc: 0.8658
Test accuracy: 86.58%
-----The CNN model is completedly trained-----
Training Time Elapsed: 0:00:13.764889
```

In [36]:
```
# Plot the Learning Curve
CNN.plotLearningCurve(history);
```



Prediction based on Training Set

In [37]:
```
y_pred = cnnModel.predict(X_test=X_train);
print(f"Number of predictions made: {len(y_pred)}");
print(f"Shape of predictions array: {y_pred.shape}");
print(f"Dimentionality of predictions array: {y_pred.ndim}");
```

```
-----Started Predicting test samples-----
19/19 [==============================] - 0s 3ms/step
-----Prediction completed-----
Prediction Time Elapsed: 0:00:00.181662
Number of predictions made: 587
Shape of predictions array: (587, 1)
Dimentionality of predictions array: 2
```

Since the predictions array is in 2-dimension, we have to inspect an element in it and perform some conversions to the array.

In [38]:
```
y_pred[random.randint(0, len(y_pred))]
```

Out[38]:
```
array([0.33085027], dtype=float32)
```

We now understood that an element is basically a number (between 0 and 1) being wrapped in a list. Therefore, we now perform the conversion to `y_pred`.

In [39]:
```
y_pred = convertPredArray(y_pred);
# Convert predictions to class labels (0 or 1) based on a threshold of 0.5
y_pred = (y_pred > 0.5).astype(int);
print(f"Number of predictions made: {len(y_pred)}");
print(f"Shape of predictions array: {y_pred.shape}");
print(f"Dimentionality of predictions array: {y_pred.ndim}");
print(f"An element in the converted predictions array now: {y_pred[random.randint(0
```

```
Number of predictions made: 587
Shape of predictions array: (587,)
Dimentionality of predictions array: 1
An element in the converted predictions array now: 1
```

In [40]:
```
# Recap
print(f"Number of actual samples in the test set: {len(X_train)}");
print(f"Number of actual labels in the test set: {len(y_train)}");
print(f"Shape of predictions array: {y_train.shape}");
```

```
Number of actual samples in the test set: 587
Number of actual labels in the test set: 587
Shape of predictions array: (587,)
```

We know that all samples have been successfully predicted from the training set. Now we display the confusion matrix regarding the prediction just made based on the Training Set.

In [41]:
```
printConfMtx(
    y_pred=y_pred,
    y_test=y_train
)
```

Out[41]:

Predicted	Yor	Cal
Actual		
Yor	265	69
Cal	41	212

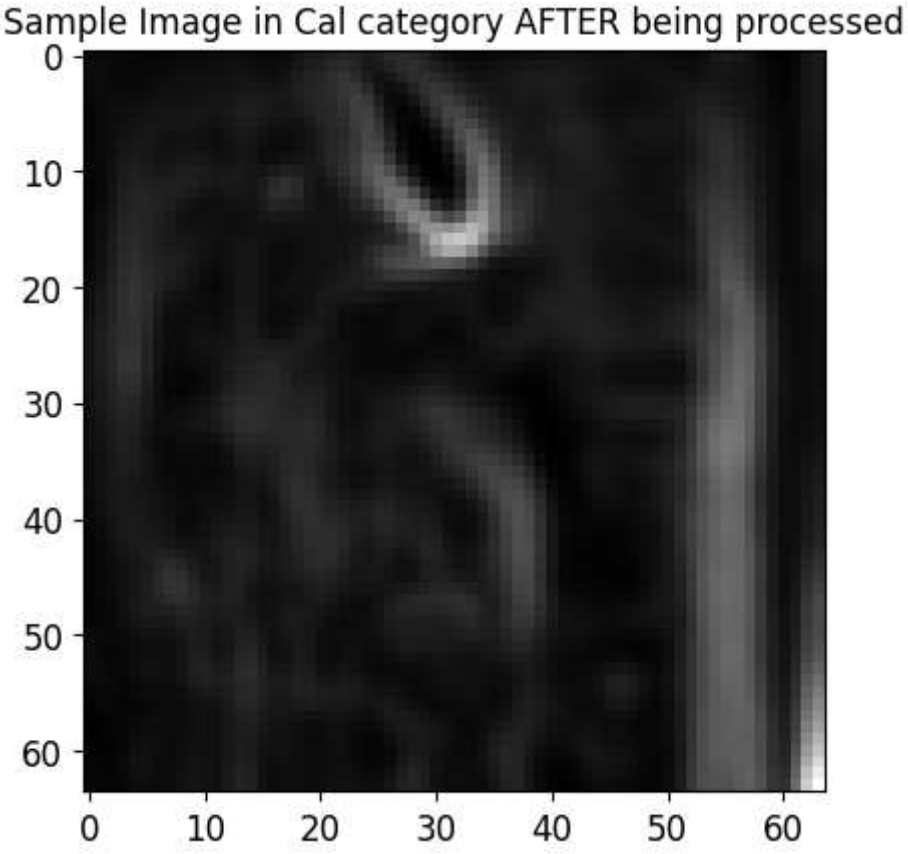
In [42]:
```
# Evaluate Metrics
# first item is for the metric score in training set; second item is for that in tr
accuracies_cnn, precisions_cnn, recalls_cnn, f1s_cnn = [], [], [], [];
accuracies_cnn.append(getAccuracy(
    y_pred=y_pred,
    y_test=y_train
));
precisions_cnn.append(getPrecision(
    y_pred=y_pred,
    y_test=y_train
));
recalls_cnn.append(getRecall(
    y_pred=y_pred,
```

```
    y_test=y_train
));
f1s_cnn.append(getF1(
    y_pred=y_pred,
    y_test=y_train
));
print(f"Accuracy of this MLP model based on Training Set: {accuracies_cnn[0]} ");
print(f"Precision of this MLP model based on Training Set: {precisions_cnn[0]} ");
print(f"Recall of this MLP model based on Training Set: {recalls_cnn[0]}");
print(f"F1 Score of this MLP model based on Training Set: {f1s_cnn[0]}");
```

```
Accuracy of this MLP model based on Training Set: 0.8126064735945485
Precision of this MLP model based on Training Set: 0.8660130718954249
Recall of this MLP model based on Training Set: 0.7934131736526946
F1 Score of this MLP model based on Training Set: 0.828125
```

Visualizing a Misclassified Image In the Training Set

In [43]:
```
visualizeMisclassified(
    X_test=X_train,
    y_test=y_train,
    y_pred=y_pred
);
```




```
Number of misclassified images for this model: 110 out of total 587 images in the se
t.
```

Prediction based on Test Set

In [44]:
```
y_pred = cnnModel.predict(X_test=X_test);
print(f"Number of predictions made: {len(y_pred)}");
print(f"Shape of predictions array: {y_pred.shape}");
print(f"Dimentionality of predictions array: {y_pred.ndim}");
```

```
-----Started Predicting test samples-----
13/13 [==============================] - 0s 3ms/step
-----Prediction completed-----
Prediction Time Elapsed: 0:00:00.134528
Number of predictions made: 395
Shape of predictions array: (395, 1)
Dimentionality of predictions array: 2
```

Since the predictions array is in 2-dimension, we have to inspect an element in it and perform some conversions to the array.

In [45]:
```
y_pred[random.randint(0, len(y_pred))]
```

Out[45]:
```
array([0.52530396], dtype=float32)
```

We now understood that an element is basically a number (between 0 and 1) being wrapped in a list. Therefore, we now perform the conversion to `y_pred`.

In [46]:
```
y_pred = convertPredArray(y_pred);
# Convert predictions to class labels (0 or 1) based on a threshold of 0.5
y_pred = (y_pred > 0.5).astype(int);
print(f"Number of predictions made: {len(y_pred)}");
print(f"Shape of predictions array: {y_pred.shape}");
print(f"Dimentionality of predictions array: {y_pred.ndim}");
print(f"An element in the converted predictions array now: {y_pred[random.randint(0
```

```
Number of predictions made: 395
Shape of predictions array: (395,)
Dimentionality of predictions array: 1
An element in the converted predictions array now: 0
```

In [47]:
```
# Recap
print(f"Number of actual samples in the test set: {len(X_test)}");
print(f"Number of actual labels in the test set: {len(y_test)}");
print(f"Shape of predictions array: {y_test.shape}");
```

```
Number of actual samples in the test set: 395
Number of actual labels in the test set: 395
Shape of predictions array: (395,)
```

We know that all samples have been successfully predicted from the training set. Now we display the confusion matrix regarding the prediction just made based on the Test Set.

In [48]:
```
printConfMtx(
    y_pred=y_pred,
    y_test=y_test
)
```

Out[48]:

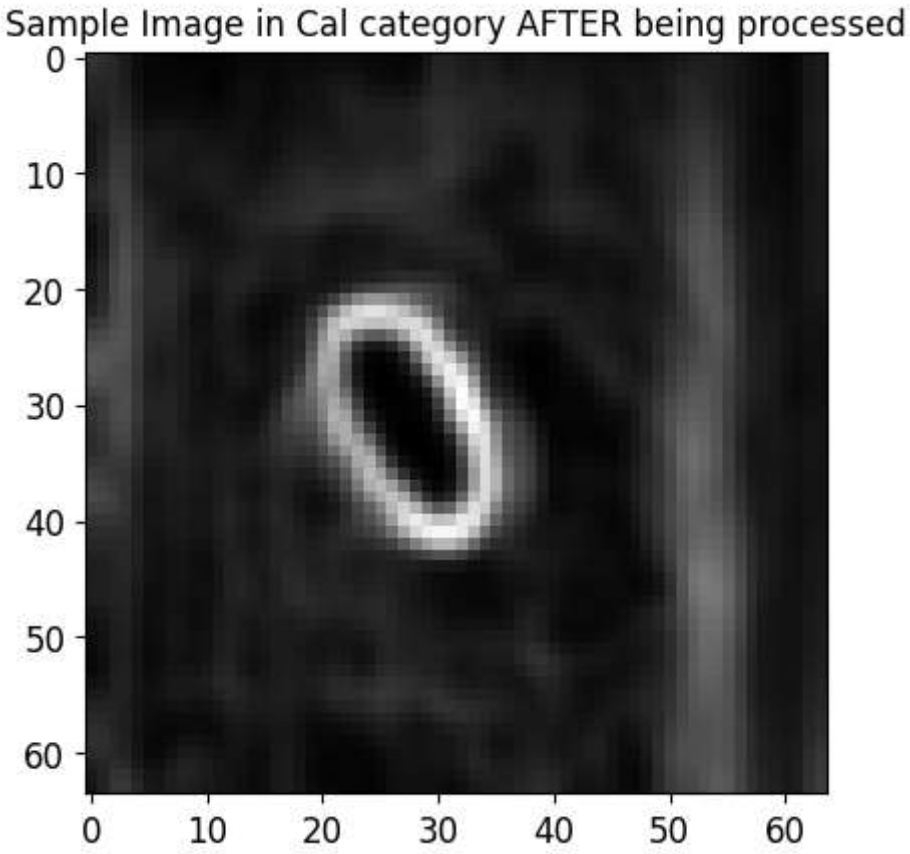
Actual	Predicted Yor	Predicted Cal
Yor	121	37
Cal	16	221

In [49]:
```
# Evaluate Metrics
# first item is for the metric score in training set; second item is for that in te
accuracies_cnn.append(getAccuracy(
    y_pred=y_pred,
    y_test=y_test
));
precisions_cnn.append(getPrecision(
    y_pred=y_pred,
    y_test=y_test
));
recalls_cnn.append(getRecall(
    y_pred=y_pred,
    y_test=y_test
));
f1s_cnn.append(getF1(
    y_pred=y_pred,
    y_test=y_test
));
print(f"Accuracy of this MLP model based on Test Set: {accuracies_cnn[-1]} ");
print(f"Precision of this MLP model based on Test Set: {precisions_cnn[-1]} ");
print(f"Recall of this MLP model based on Test Set: {recalls_cnn[-1]} ");
print(f"F1 Score of this MLP model based on Test Set: {f1s_cnn[-1]} ");
```

```
Accuracy of this MLP model based on Test Set: 0.8658227848101265
Precision of this MLP model based on Test Set: 0.8832116788321168
Recall of this MLP model based on Test Set: 0.7658227848101266
F1 Score of this MLP model based on Test Set: 0.8203389830508474
```

Visualizing a Misclassified Image in the Test Set

In [50]:
```
visualizeMisclassified(
    X_test=X_test,
    y_test=y_test,
    y_pred=y_pred
);
```




```
Number of misclassified images for this model: 53 out of total 395 images in the se
t.
```

Analysis and Summary

Let's summarize the metrics from the MLP model.

In [51]:
```
print(f"Prediction Accuracy based on training set: {accuracies_mlp[0]}");
print(f"Prediction Precision based on training set: {precisions_mlp[0]}");
print(f"Prediction Recall based on training set: {recalls_mlp[0]}");
print(f"Prediction F1 score based on training set: {f1s_mlp[0]}");
print();
print(f"Prediction Accuracy based on test set: {accuracies_mlp[-1]}");
print(f"Prediction Precision based on test set: {precisions_mlp[-1]}");
print(f"Prediction Recall based on test set: {recalls_mlp[-1]}");
print(f"Prediction F1 score based on test set: {f1s_mlp[-1]}");
print();
mean_accuracy = np.mean(accuracies_mlp);
mean_precision = np.mean(precisions_mlp);
mean_recall = np.mean(recalls_mlp);
mean_f1 = np.mean(f1s_mlp);
print(f"And therefore, the mean accuracy of the MLP model across every set of data
print(f"And therefore, the mean precision of the MLP model across every set of data
print(f"And therefore, the mean recall of the MLP model across every set of data =
print(f"And therefore, the mean F1 score of the MLP model across every set of data
```

```
Prediction Accuracy based on training set: 0.5689948892674617
Prediction Precision based on training set: 0.5689948892674617
Prediction Recall based on training set: 1.0
Prediction F1 score based on training set: 0.7252985884907709

Prediction Accuracy based on test set: 0.4
Prediction Precision based on test set: 0.4
Prediction Recall based on test set: 1.0
Prediction F1 score based on test set: 0.5714285714285714

And therefore, the mean accuracy of the MLP model across every set of data = 0.48449
744463373084
And therefore, the mean precision of the MLP model across every set of data = 0.4844
9744463373084
And therefore, the mean recall of the MLP model across every set of data = 1.0
And therefore, the mean F1 score of the MLP model across every set of data = 0.64836
35799596712
```

Since the scores in the training set are generally higher than those evaluated from the test set, there exists some overfitting issue, which means the training data generalizes the MLP model too good. But this isn't a huge difference, and therefore, overfitting issue is not too serious.

Let's also summarize the metrics from the CNN model.

In [52]:
```
print(f"Prediction Accuracy based on training set: {accuracies_cnn[0]}");
print(f"Prediction Precision based on training set: {precisions_cnn[0]}");
print(f"Prediction Recall based on training set: {recalls_cnn[0]}");
print(f"Prediction F1 score based on training set: {f1s_cnn[0]}");
print();
print(f"Prediction Accuracy based on test set: {accuracies_cnn[-1]}");
print(f"Prediction Precision based on test set: {precisions_cnn[-1]}");
print(f"Prediction Recall based on test set: {recalls_cnn[-1]}");
print(f"Prediction F1 score based on test set: {f1s_cnn[-1]}");
print();
mean_accuracy = np.mean(accuracies_cnn);
mean_precision = np.mean(precisions_cnn);
mean_recall = np.mean(recalls_cnn);
mean_f1 = np.mean(f1s_cnn);
print(f"And therefore, the mean accuracy of the CNN model across every set of data
print(f"And therefore, the mean precision of the CNN model across every set of data
print(f"And therefore, the mean recall of the CNN model across every set of data =
print(f"And therefore, the mean F1 score of the CNN model across every set of data
```

```
Prediction Accuracy based on training set: 0.8126064735945485
Prediction Precision based on training set: 0.8660130718954249
Prediction Recall based on training set: 0.7934131736526946
Prediction F1 score based on training set: 0.828125

Prediction Accuracy based on test set: 0.8658227848101265
Prediction Precision based on test set: 0.8832116788321168
Prediction Recall based on test set: 0.7658227848101266
Prediction F1 score based on test set: 0.8203389830508474

And therefore, the mean accuracy of the CNN model across every set of data = 0.83921
46292023375
And therefore, the mean precision of the CNN model across every set of data = 0.8746
123753637709
And therefore, the mean recall of the CNN model across every set of data = 0.7796179
792314106
And therefore, the mean F1 score of the CNN model across every set of data = 0.82423
19915254237
```

Since the scores in the training set are very similar to those evaluated from the test set, there isn't any potential overfitting nor underfitting issues with the model to the given data.

Other Observations:

- Training Speed of the MLP model = 0:00:02.096241
- Training Speed of the CNN model = 0:00:13.764889
- Prediction Speed of the MLP model = 0:00:00.079966
- Prediction Speed of the CNN model = 0:00:00.134528
- Test Accuracy evaluated while training the MLP model = 40.0%
- Test Accuracy evaluated while training the CNN model = 86.58%
- Loss of MLP model = 0.6983
- Loss of CNN model = 0.3198

Comparison:

- Except the mean recall, all other mean scores from the CNN model are higher than that from the MLP model.
- The CNN model took longer time to train.
- The CNN model took longer time to make predictions.
- The CNN model seems to have higher test accuracy.
- The CNN model seems to have lower loss rate.

Conclusion:

- Therefore, despite more time taken to train the model or to make predictions, the CNN generally performs better than the MLP model.