

```

{
  "cells": [
    {
      "cell_type": "markdown",
      "metadata": {},
      "source": [
        "# ECG arrhythmia classification using CNN"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 1,
      "metadata": {},
      "outputs": [
        {
          "data": {
            "text/plain": [
              "'/home/wsuser/work'"
            ]
          },
          "execution_count": 1,
          "metadata": {},
          "output_type": "execute_result"
        }
      ],
      "source": [
        "pwd"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 5,
      "metadata": {},
      "outputs": [
        {
          "name": "stdout",
          "output_type": "stream",
          "text": [
            "Requirement already satisfied: keras in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (2.9.0)\n",
            "Requirement already satisfied: tensorflow in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (2.9.1)\n",
            "Requirement already satisfied: six>=1.12.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.15.0)\n",
            "Requirement already satisfied: astunparse>=1.6.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (1.6.3)\n",
            "Requirement already satisfied: keras<2.10.0,>=2.9.0rc0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (2.9.0)\n",
            "Requirement already satisfied: libclang>=13.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (14.0.1)\n",
            "Requirement already satisfied: tensorboard<2.10,>=2.9 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (2.9.0)\n",
            "Requirement already satisfied: setuptools in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow) (58.0.4)\n",

```

"Requirement already satisfied: keras-preprocessing>=1.1.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(1.1.2)\n",
"Requirement already satisfied: opt-einsum>=2.3.2 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(3.3.0)\n",
"Requirement already satisfied: absl-py>=1.0.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(1.1.0)\n",
"Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1
in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(0.23.1)\n",
"Requirement already satisfied: typing-extensions>=3.6.6 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(3.7.4.3)\n",
"Requirement already satisfied: google-pasta>=0.1.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(0.2.0)\n",
"Requirement already satisfied: tensorflow-
estimator<2.10.0,>=2.9.0rc0 in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from tensorflow) (2.9.0)\n",
"Requirement already satisfied: protobuf<3.20,>=3.9.2 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(3.19.1)\n",
"Requirement already satisfied: grpcio<2.0,>=1.24.3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(1.42.0)\n",
"Requirement already satisfied: termcolor>=1.1.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(1.1.0)\n",
"Requirement already satisfied: numpy>=1.20 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(1.20.3)\n",
"Requirement already satisfied: gast<=0.4.0,>=0.2.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(0.4.0)\n",
"Requirement already satisfied: h5py>=2.9.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(3.2.1)\n",
"Requirement already satisfied: flatbuffers<2,>=1.12 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(1.12)\n",
"Requirement already satisfied: wrapt>=1.11.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from tensorflow)
(1.12.1)\n",
"Requirement already satisfied: packaging in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from tensorflow) (21.3)\n",
"Requirement already satisfied: wheel<1.0,>=0.23.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
astunparse>=1.6.0->tensorflow) (0.37.0)\n",
"Requirement already satisfied: werkzeug>=1.0.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
tensorboard<2.10,>=2.9->tensorflow) (2.0.2)\n",
"Requirement already satisfied: google-auth<3,>=1.6.3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
tensorboard<2.10,>=2.9->tensorflow) (1.23.0)\n",
"Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
tensorboard<2.10,>=2.9->tensorflow) (1.6.0)\n",

```

        "Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0
in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
tensorboard<2.10,>=2.9->tensorflow) (0.6.1)\n",
        "Requirement already satisfied: requests<3,>=2.21.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
tensorboard<2.10,>=2.9->tensorflow) (2.26.0)\n",
        "Requirement already satisfied: markdown>=2.6.8 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
tensorboard<2.10,>=2.9->tensorflow) (3.3.3)\n",
        "Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
tensorboard<2.10,>=2.9->tensorflow) (0.4.4)\n",
        "Requirement already satisfied: rsa<5,>=3.1.4 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from google-
auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (4.7.2)\n",
        "Requirement already satisfied: pyasn1-modules>=0.2.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from google-
auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (0.2.8)\n",
        "Requirement already satisfied: cachetools<5.0,>=2.0.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from google-
auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (4.2.2)\n",
        "Requirement already satisfied: requests-oauthlib>=0.7.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from google-auth-
oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9->tensorflow) (1.3.0)\n",
        "Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pyasn1-
modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow)
(0.4.8)\n",
        "Requirement already satisfied: idna<4,>=2.5 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (3.3)\n",
        "Requirement already satisfied: charset-normalizer~=2.0.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (2.0.4)\n",
        "Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (1.26.7)\n",
        "Requirement already satisfied: certifi>=2017.4.17 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (2022.5.18.1)\n",
        "Requirement already satisfied: oauthlib>=3.0.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests-
oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9-
>tensorflow) (3.2.0)\n",
        "Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from packaging-
>tensorflow) (3.0.4)\n"
    ]
}
],
"source": [
    "!pip install keras\n",
    "!pip install tensorflow"
]
},
{
    "cell_type": "code",
    "execution_count": 6,
    "metadata": {},
    "outputs": [
        {

```

```

    "data": {
      "text/plain": [
        "'2.9.0'"
      ]
    },
    "execution_count": 6,
    "metadata": {},
    "output_type": "execute_result"
  },
  "source": [
    "import keras\n",
    "keras.__version__"
  ]
},
{
  "cell_type": "code",
  "execution_count": 7,
  "metadata": {},
  "outputs": [
    {
      "data": {
        "text/plain": [
          "'2.9.1'"
        ]
      },
      "execution_count": 7,
      "metadata": {},
      "output_type": "execute_result"
    }
  ],
  "source": [
    "import tensorflow\n",
    "tensorflow.__version__"
  ]
},
{
  "cell_type": "markdown",
  "metadata": {},
  "source": [
    "### Importing Neccessary Libraries"
  ]
},
{
  "cell_type": "code",
  "execution_count": 8,
  "metadata": {},
  "outputs": [],
  "source": [
    "import numpy as np#used for numerical analysis\n",
    "import tensorflow #open source used for both ML and DL for\n",
    "computation\n",
    "from tensorflow.keras.models import Sequential #it is a plain stack of\n",
    "layers\n",
    "from tensorflow.keras import layers #A layer consists of a tensor-in\n",
    "tensor-out computation function\n",
    "#Dense layer is the regular deeply connected neural network layer\n",
    "from tensorflow.keras.layers import Dense,Flatten\n",
    "#Falppen-used fot flattening the input or change the dimension\n",
    "from tensorflow.keras.layers import Conv2D,MaxPooling2D #Convolutional\n",
    "layer\n",

```

```

    "#MaxPooling2D-for downsampling the image\n",
    "from keras.preprocessing.image import ImageDataGenerator\n"
]
},
{
    "cell_type": "markdown",
    "metadata": {},
    "source": [
        "### Image Data Agumentation"
    ]
},
{
    "cell_type": "code",
    "execution_count": 9,
    "metadata": {},
    "outputs": [],
    "source": [
        "#setting parameter for Image Data agumentation to the traing data\n",

"train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range
=0.2,horizontal_flip=True)"
    ]
},
{
    "cell_type": "code",
    "execution_count": 10,
    "metadata": {},
    "outputs": [],
    "source": [
        "#Image Data agumentation to the testing data\n",
        "test_datagen=ImageDataGenerator(rescale=1./255)"
    ]
},
{
    "cell_type": "markdown",
    "metadata": {},
    "source": [
        "### Loading our data and performing data agumentation"
    ]
},
{
    "cell_type": "code",
    "execution_count": 11,
    "metadata": {},
    "outputs": [],
    "source": [
        "\n",
        "import os, types\n",
        "import pandas as pd\n",
        "from botocore.client import Config\n",
        "import ibm_boto3\n",
        "\n",
        "def __iter__(self): return 0\n",
        "\n",
        "# @hidden_cell\n",
        "# The following code accesses a file in your IBM Cloud Object Storage.
        It includes your credentials.\n",
        "# You might want to remove those credentials before you share the
        notebook.\n",
        "\n",
        "if os.environ.get('RUNTIME_ENV_LOCATION_TYPE') == 'external':\n",

```

```

        "        endpoint_69c05974e5c84795a978662af2736fc1 = 'https://s3.us.cloud-
object-storage.appdomain.cloud'\n",
        "else:\n",
        "        endpoint_69c05974e5c84795a978662af2736fc1 =
'https://s3.private.us.cloud-object-storage.appdomain.cloud'\n",
        "\n",
        "client_69c05974e5c84795a978662af2736fc1 =
ibm_boto3.client(service_name='s3',\n",
        "        ibm_api_key_id='KNnyzgYsEAWBFEGgzWg_6j5CyAcjpaOlCPz3gd9KEdr_',\n",
        "        ibm_auth_endpoint=\"https://iam.cloud.ibm.com/oidc/token\",\n",
        "        config=Config(signature_version='oauth'),\n",
        "        endpoint_url=endpoint_69c05974e5c84795a978662af2736fc1)\n",
        "\n",
        "streaming_body_1 =
client_69c05974e5c84795a978662af2736fc1.get_object(Bucket='ecgimagebasedhea
rtbeatclassificat-donotdelete-pr-l2ugdciyflqayf',
Key='data.zip')['Body']\n",
        "\n",
        "# Your data file was loaded into a botocore.response.StreamingBody
object.\n",
        "# Please read the documentation of ibm_boto3 and pandas to learn more
about the possibilities to load the data.\n",
        "# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-
python/\n",
        "# pandas documentation: http://pandas.pydata.org/\n"
    ]
},
{
    "cell_type": "code",
    "execution_count": 12,
    "metadata": {},
    "outputs": [
        {
            "name": "stdout",
            "output_type": "stream",
            "text": [
                '['data/test/', 'data/test/Left Bundle Branch Block/',
                'data/test/Left Bundle Branch Block/fig_5897.png', 'data/test/Left Bundle
                Branch Block/fig_5898.png', 'data/test/Left Bundle Branch
                Block/fig_5899.png', 'data/test/Left Bundle Branch Block/fig_5900.png',
                'data/test/Left Bundle Branch Block/fig_5901.png', 'data/test/Left Bundle
                Branch Block/fig_5902.png', 'data/test/Left Bundle Branch
                Block/fig_5903.png', 'data/test/Left Bundle Branch Block/fig_5904.png',
                'data/test/Left Bundle Branch Block/fig_5905.png', 'data/test/Left Bundle
                Branch Block/fig_5906.png', 'data/test/Left Bundle Branch
                Block/fig_5907.png', 'data/test/Left Bundle Branch Block/fig_5908.png',
                'data/test/Left Bundle Branch Block/fig_5909.png', 'data/test/Left Bundle
                Branch Block/fig_5910.png', 'data/test/Left Bundle Branch
                Block/fig_5911.png', 'data/test/Left Bundle Branch Block/fig_5912.png',
                'data/test/Left Bundle Branch Block/fig_5913.png', 'data/test/Left Bundle
                Branch Block/fig_5914.png', 'data/test/Left Bundle Branch
                Block/fig_5915.png', 'data/test/Left Bundle Branch Block/fig_5916.png',
                'data/test/Left Bundle Branch Block/fig_5917.png', 'data/test/Left Bundle
                Branch Block/fig_5918.png', 'data/test/Left Bundle Branch
                Block/fig_5919.png', 'data/test/Left Bundle Branch Block/fig_5920.png',
                'data/test/Left Bundle Branch Block/fig_5921.png', 'data/test/Left Bundle
                Branch Block/fig_5922.png', 'data/test/Left Bundle Branch
                Block/fig_5923.png', 'data/test/Left Bundle Branch Block/fig_5924.png',
                'data/test/Left Bundle Branch Block/fig_5925.png', 'data/test/Left Bundle
                Branch Block/fig_5926.png', 'data/test/Left Bundle Branch
                Block/fig_5927.png', 'data/test/Left Bundle Branch Block/fig_5928.png',

```

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

```

'data/train/Ventricular Fibrillation/VFEfig_67.png',
'data/train/Ventricular Fibrillation/VFEfig_68.png',
'data/train/Ventricular Fibrillation/VFEfig_69.png',
'data/train/Ventricular Fibrillation/VFEfig_70.png',
'data/train/Ventricular Fibrillation/VFEfig_71.png',
'data/train/Ventricular Fibrillation/VFEfig_72.png',
'data/train/Ventricular Fibrillation/VFEfig_73.png',
'data/train/Ventricular Fibrillation/VFEfig_74.png',
'data/train/Ventricular Fibrillation/VFEfig_75.png',
'data/train/Ventricular Fibrillation/VFEfig_76.png',
'data/train/Ventricular Fibrillation/VFEfig_77.png',
'data/train/Ventricular Fibrillation/VFEfig_78.png',
'data/train/Ventricular Fibrillation/VFEfig_79.png',
'data/train/Ventricular Fibrillation/VFEfig_80.png',
'data/train/Ventricular Fibrillation/VFEfig_81.png',
'data/train/Ventricular Fibrillation/VFEfig_82.png',
'data/train/Ventricular Fibrillation/VFEfig_83.png',
'data/train/Ventricular Fibrillation/VFEfig_84.png',
'data/train/Ventricular Fibrillation/VFEfig_85.png',
'data/train/Ventricular Fibrillation/VFEfig_86.png',
'data/train/Ventricular Fibrillation/VFEfig_87.png',
'data/train/Ventricular Fibrillation/VFEfig_88.png',
'data/train/Ventricular Fibrillation/VFEfig_89.png',
'data/train/Ventricular Fibrillation/VFEfig_90.png',
'data/train/Ventricular Fibrillation/VFEfig_91.png',
'data/train/Ventricular Fibrillation/VFEfig_92.png',
'data/train/Ventricular Fibrillation/VFEfig_93.png',
'data/train/Ventricular Fibrillation/VFEfig_94.png',
'data/train/Ventricular Fibrillation/VFEfig_95.png',
'data/train/Ventricular Fibrillation/VFEfig_96.png',
'data/train/Ventricular Fibrillation/VFEfig_97.png',
'data/train/Ventricular Fibrillation/VFEfig_98.png',
'data/train/Ventricular Fibrillation/VFEfig_99.png']\n"
    ]
    }
],
"source": [
    "from io import BytesIO\n",
    "import zipfile\n",
    "unzip = zipfile.ZipFile(BytesIO(streaming_body_1.read()), 'r')\n",
    "file_paths=unzip.namelist()\n",
    "print(file_paths)\n",
    "for path in file_paths:\n",
    "    unzip.extract(path)"
]
},
{
    "cell_type": "code",
    "execution_count": 13,
    "metadata": {},
    "outputs": [
        {
            "data": {
                "text/plain": [
                    "'/home/wsuser/work'"
                ]
            },
            "execution_count": 13,
            "metadata": {},
            "output_type": "execute_result"
        }
    ]
}

```



```

],
"source": [
    "pwd"
]
},
{
    "cell_type": "code",
    "execution_count": 14,
    "metadata": {},
    "outputs": [],
    "source": [
        "import os\n",
        "filenames = os.listdir('/home/wsuser/work/data')"
    ]
},
{
    "cell_type": "code",
    "execution_count": 15,
    "metadata": {},
    "outputs": [
        {
            "name": "stdout",
            "output_type": "stream",
            "text": [
                "Found 15341 images belonging to 6 classes.\n",
                "Found 6825 images belonging to 6 classes.\n"
            ]
        }
    ],
    "source": [
        "#performing data agumentation to train data\n",

"x_train=train_datagen.flow_from_directory('/home/wsuser/work/data/train',t
arget_size=(64,64),batch_size=32,class_mode='categorical')\n",
        "#performing data agumentation to test data\n",

"x_test=test_datagen.flow_from_directory('/home/wsuser/work/data/test',targ
et_size=(64,64),batch_size=32,class_mode='categorical')"
    ]
},
{
    "cell_type": "code",
    "execution_count": 16,
    "metadata": {},
    "outputs": [
        {
            "name": "stdout",
            "output_type": "stream",
            "text": [
                "{ 'Left Bundle Branch Block': 0, 'Normal': 1, 'Premature Atrial
Contraction': 2, 'Premature Ventricular Contractions': 3, 'Right Bundle
Branch Block': 4, 'Ventricular Fibrillation': 5}\n"
            ]
        }
    ],
    "source": [
        "print(x_train.class_indices)#checking the number of classes"
    ]
},
{
    "cell_type": "code",

```

```

"execution_count": 17,
"metadata": {},
"outputs": [
  {
    "data": {
      "text/plain": [
        "Counter({0: 504, 1: 7346, 2: 2054, 3: 2759, 4: 2239, 5: 439})"
      ]
    },
    "execution_count": 17,
    "metadata": {},
    "output_type": "execute_result"
  }
],
"source": [
  "from collections import Counter as c\n",
  "c(x_train.labels)"
]
},
{
  "cell_type": "markdown",
  "metadata": {},
  "source": [
    "### Creating the model"
  ]
},
{
  "cell_type": "code",
  "execution_count": 18,
  "metadata": {},
  "outputs": [
    {
      "name": "stderr",
      "output_type": "stream",
      "text": [
        "2022-06-03 15:08:02.006929: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not
load dynamic library 'libcuda.so.1'; dLError: libcuda.so.1: cannot open
shared object file: No such file or directory; LD_LIBRARY_PATH:
/opt/ibm/dsdriver/lib:/opt/oracle/lib:/opt/conda/envs/Python-
3.9/lib/python3.9/site-packages/tensorflow\n",
        "2022-06-03 15:08:02.006997: W
tensorflow/stream_executor/cuda/cuda_driver.cc:269] failed call to cuInit:
UNKNOWN ERROR (303)\n"
      ]
    }
  ],
  "source": [
    "# create model\n",
    "model=Sequential()\n",
    "# adding model layer\n",

    "model.add(Conv2D(32, (3,3), input_shape=(64,64,3), activation='relu'))#convol
utional layer\n",
    "model.add(MaxPooling2D(pool_size=(2,2))) #MaxPooling2D-for
downsampling the input\n",
    "\n",
    "model.add(Conv2D(32, (3,3), activation='relu'))\n",
    "model.add(MaxPooling2D(pool_size=(2,2)))\n",
    "\n",
    "model.add(Flatten())#flatten the dimension of the image\n",

```

```

        "model.add(Dense(32))#deeply connected neural network layers.\n",
        "model.add(Dense(6,activation='softmax'))#output layer with 6
neurons\n"
    ]
    },
    {
        "cell_type": "code",
        "execution_count": 19,
        "metadata": {},
        "outputs": [
            {
                "name": "stdout",
                "output_type": "stream",
                "text": [
                    "Model: \"sequential\"\n\n",
                    "
                    _____\n",
                    " Layer (type)                Output Shape          Param #
                    \n",
                    "=====
                    " conv2d (Conv2D)              (None, 62, 62, 32)    896
                    \n",
                    "
                    \n",
                    " max_pooling2d (MaxPooling2D  (None, 31, 31, 32)    0
                    \n",
                    " )
                    \n",
                    "
                    \n",
                    " conv2d_1 (Conv2D)              (None, 29, 29, 32)    9248
                    \n",
                    "
                    \n",
                    " max_pooling2d_1 (MaxPooling  (None, 14, 14, 32)    0
                    \n",
                    " 2D)
                    \n",
                    "
                    \n",
                    " flatten (Flatten)             (None, 6272)          0
                    \n",
                    "
                    \n",
                    " dense (Dense)                 (None, 32)            200736
                    \n",
                    "
                    \n",
                    " dense_1 (Dense)               (None, 6)             198
                    \n",
                    "
                    \n",
                    "=====
                    "Total params: 211,078\n",
                    "Trainable params: 211,078\n",
                    "Non-trainable params: 0\n",
                    "
                    _____\n",
                ]
            }
        ]
    }
}

```

```

],
"source": [
    "model.summary()#summary of our model"
]
},
{
    "cell_type": "markdown",
    "metadata": {},
    "source": [
        "### Compiling the model"
    ]
},
{
    "cell_type": "code",
    "execution_count": 20,
    "metadata": {},
    "outputs": [],
    "source": [
        "# Compile model\n",
        "model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['a
ccuracy'])"
    ]
},
{
    "cell_type": "markdown",
    "metadata": {},
    "source": [
        "### Fitting the model"
    ]
},
{
    "cell_type": "code",
    "execution_count": 21,
    "metadata": {},
    "outputs": [
        {
            "name": "stderr",
            "output_type": "stream",
            "text": [
                "/tmp/wsuser/ipykernel_2832/1916141677.py:2: UserWarning:
`Model.fit_generator` is deprecated and will be removed in a future
version. Please use `Model.fit`, which supports generators.\n",
                "    model.fit_generator(generator=x_train,steps_per_epoch =
len(x_train),\n"
            ]
        },
        {
            "name": "stdout",
            "output_type": "stream",
            "text": [
                "Epoch 1/10\n",
                "480/480 [=====] - 97s 201ms/step - loss:
0.7450 - accuracy: 0.7491 - val_loss: 0.4413 - val_accuracy: 0.8504\n",
                "Epoch 2/10\n",
                "480/480 [=====] - 96s 201ms/step - loss:
0.2724 - accuracy: 0.9186 - val_loss: 0.4429 - val_accuracy: 0.8809\n",
                "Epoch 3/10\n",
                "480/480 [=====] - 97s 201ms/step - loss:
0.2204 - accuracy: 0.9347 - val_loss: 0.2756 - val_accuracy: 0.9207\n",
                "Epoch 4/10\n",

```

```

    "480/480 [=====] - 97s 201ms/step - loss:
0.1896 - accuracy: 0.9435 - val_loss: 0.2367 - val_accuracy: 0.9314\n",
    "Epoch 5/10\n",
    "480/480 [=====] - 96s 201ms/step - loss:
0.1637 - accuracy: 0.9497 - val_loss: 0.2607 - val_accuracy: 0.9174\n",
    "Epoch 6/10\n",
    "480/480 [=====] - 96s 200ms/step - loss:
0.1526 - accuracy: 0.9544 - val_loss: 0.2423 - val_accuracy: 0.9263\n",
    "Epoch 7/10\n",
    "480/480 [=====] - 96s 199ms/step - loss:
0.1366 - accuracy: 0.9567 - val_loss: 0.2960 - val_accuracy: 0.9169\n",
    "Epoch 8/10\n",
    "480/480 [=====] - 97s 202ms/step - loss:
0.1232 - accuracy: 0.9608 - val_loss: 0.3316 - val_accuracy: 0.9078\n",
    "Epoch 9/10\n",
    "480/480 [=====] - 97s 201ms/step - loss:
0.1255 - accuracy: 0.9626 - val_loss: 0.2454 - val_accuracy: 0.9297\n",
    "Epoch 10/10\n",
    "480/480 [=====] - 95s 199ms/step - loss:
0.1174 - accuracy: 0.9638 - val_loss: 0.3265 - val_accuracy: 0.9182\n"
]
},
{
    "data": {
        "text/plain": [
            "<keras.callbacks.History at 0x7fc31dddeff70>"
        ]
    },
    "execution_count": 21,
    "metadata": {},
    "output_type": "execute_result"
}
],
"source": [
    "# Fit the model \n",
    "model.fit_generator(generator=x_train, steps_per_epoch =
len(x_train), \n",
    "                    epochs=10, validation_data=x_test, validation_steps
= len(x_test))"
]
},
{
    "cell_type": "code",
    "execution_count": 22,
    "metadata": {},
    "outputs": [],
    "source": [
        "# model.fit_generator(x_train, epochs=10, validation_data=x_test)"
    ]
},
{
    "cell_type": "markdown",
    "metadata": {},
    "source": [
        "### Saving our model"
    ]
},
{
    "cell_type": "code",
    "execution_count": 23,
    "metadata": {},

```

```

"outputs": [],
"source": [
    "# Save the model\n",
    "from tensorflow.keras.models import load_model\n",
    "model.save('ECG.h5') "
]
},
{
    "cell_type": "code",
    "execution_count": 24,
    "metadata": {},
    "outputs": [
        {
            "name": "stdout",
            "output_type": "stream",
            "text": [
                "ECG.h5\r\n"
            ]
        }
    ],
    "source": [
        "!tar -zcvf ECG-Image-based-heartbeat-classification-model_new.tgz
ECG.h5"
    ]
},
{
    "cell_type": "code",
    "execution_count": 25,
    "metadata": {},
    "outputs": [
        {
            "name": "stdout",
            "output_type": "stream",
            "text": [
                "\u001b[0m\u001b[01;34mdata\u001b[0m\r\n",
                "ECG.h5\r\n",
                "ECG-Image-based-heartbeat-classification-model_new.tgz\r\n"
            ]
        }
    ],
    "source": [
        "ls -l"
    ]
},
{
    "cell_type": "code",
    "execution_count": 26,
    "metadata": {},
    "outputs": [
        {
            "name": "stdout",
            "output_type": "stream",
            "text": [
                "Requirement already satisfied: watson-machine-learning-client in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.391)\n",
                "Requirement already satisfied: urllib3 in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(1.26.7)\n",
                "Requirement already satisfied: boto3 in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(1.18.21)\n",

```

```

    "Requirement already satisfied: tqdm in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(4.62.3)\n",
    "Requirement already satisfied: ibm-cos-sdk in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-
machine-learning-client) (2.11.0)\n",
    "Requirement already satisfied: requests in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(2.26.0)\n",
    "Requirement already satisfied: tabulate in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(0.8.9)\n",
    "Requirement already satisfied: pandas in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(1.3.4)\n",
    "Requirement already satisfied: lomond in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(0.3.3)\n",
    "Requirement already satisfied: certifi in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from watson-machine-learning-client)
(2022.5.18.1)\n",
    "Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-
machine-learning-client) (0.10.0)\n",
    "Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-
machine-learning-client) (0.5.0)\n",
    "Requirement already satisfied: botocore<1.22.0,>=1.21.21 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-
machine-learning-client) (1.21.41)\n",
    "Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from
botocore<1.22.0,>=1.21.21->boto3->watson-machine-learning-client)
(2.8.2)\n",
    "Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-
3.9/lib/python3.9/site-packages (from python-dateutil<3.0.0,>=2.1-
>botocore<1.22.0,>=1.21.21->boto3->watson-machine-learning-client)
(1.15.0)\n",
    "Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk-
>watson-machine-learning-client) (2.11.0)\n",
    "Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk-
>watson-machine-learning-client) (2.11.0)\n",
    "Requirement already satisfied: idna<4,>=2.5 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests-
>watson-machine-learning-client) (3.3)\n",
    "Requirement already satisfied: charset-normalizer~=2.0.0 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests-
>watson-machine-learning-client) (2.0.4)\n",
    "Requirement already satisfied: pytz>=2017.3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas-
>watson-machine-learning-client) (2021.3)\n",
    "Requirement already satisfied: numpy>=1.17.3 in
/opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas-
>watson-machine-learning-client) (1.20.3)\n"
]
}
],
"source": [
"!pip install watson-machine-learning-client --upgrade"
]

```

```

    ]
  },
  {
    "cell_type": "code",
    "execution_count": 27,
    "metadata": {},
    "outputs": [],
    "source": [
      "# Replace the credentials that you got from watson machine learning
      service\n",
      "from ibm_watson_machine_learning import APIClient\n",
      "wml_credentials = {\n",
      "    \"url\": \"https://us-
      south.ml.cloud.ibm.com\", \n",
      "    \"apikey\": \"WpWfHtY_VXAXAgD4uzxBAb00C2FJstDEVyb3oXklUaRm\"\n",
      "    }\n",
      "client = APIClient(wml_credentials)\n"
    ]
  },
  {
    "cell_type": "code",
    "execution_count": 28,
    "metadata": {},
    "outputs": [],
    "source": [
      "client = APIClient(wml_credentials)"
    ]
  },
  {
    "cell_type": "code",
    "execution_count": 29,
    "metadata": {},
    "outputs": [],
    "source": [
      "def guid_from_space_name(client, space_name):\n",
      "    space = client.spaces.get_details()\n",
      "    #print(space)\n",
      "    return(next(item for item in space['resources'] if
      item['entity']['name'] == space_name)['metadata']['id'])"
    ]
  },
  {
    "cell_type": "code",
    "execution_count": 30,
    "metadata": {},
    "outputs": [
      {
        "name": "stdout",
        "output_type": "stream",
        "text": [
          "Space UID = 26b6a24d-f745-4d09-b456-8f6dfd7d9ca6\n"
        ]
      }
    ],
    "source": [
      "space_uid = guid_from_space_name(client, 'image_classification')\n",
      "print(\"Space UID = \" + space_uid)"
    ]
  },
  {

```



```

"cell_type": "code",
"execution_count": 31,
"metadata": {},
"outputs": [
  {
    "data": {
      "text/plain": [
        "'SUCCESS'"
      ]
    },
    "execution_count": 31,
    "metadata": {},
    "output_type": "execute_result"
  }
],
"source": [
  "client.set.default_space(space_uid)"
]
},
{
  "cell_type": "code",
  "execution_count": 32,
  "metadata": {},
  "outputs": [
    {
      "name": "stdout",
      "output_type": "stream",
      "text": [
        "-----\n",
- ----\n",
        "NAME                                ASSET_ID
TYPE\n",
        "default_py3.6                      0062b8c9-8b7d-44a0-a9b9-
46c416adcbd9 base\n",
        "kernel-spark3.2-scala2.12          020d69ce-7ac1-5e68-ac1a-
31189867356a base\n",
        "pytorch-onnx_1.3-py3.7-edt         069ea134-3346-5748-b513-
49120e15d288 base\n",
        "scikit-learn_0.20-py3.6            09c5a1d0-9c1e-4473-a344-
eb7b665ff687 base\n",
        "spark-mllib_3.0-scala_2.12         09f4cff0-90a7-5899-b9ed-
1ef348aebdee base\n",
        "pytorch-onnx_rt22.1-py3.9          0b848dd4-e681-5599-be41-
b5f6fccc6471 base\n",
        "ai-function_0.1-py3.6              0cdb0f1e-5376-4f4d-92dd-
da3b69aa9bda base\n",
        "shiny-r3.6                         0e6e79df-875e-4f24-8ae9-
62dcc2148306 base\n",
        "tensorflow_2.4-py3.7-horovod       1092590a-307d-563d-9b62-
4eb7d64b3f22 base\n",
        "pytorch_1.1-py3.6                 10ac12d6-6b30-4ccd-8392-
3e922c096a92 base\n",
        "tensorflow_1.15-py3.6-ddl          111e41b3-de2d-5422-a4d6-
bf776828c4b7 base\n",
        "runtime-22.1-py3.9                 12b83a17-24d8-5082-900f-
0ab31fbfd3cb base\n",
        "scikit-learn_0.22-py3.6            154010fa-5b3b-4ac1-82af-
4d5ee5abbc85 base\n",
        "default_r3.6                      1b70aec3-ab34-4b87-8aa0-
a4a3c8296a36 base\n",

```

"pytorch-onnx_1.3-py3.6 39c3880dbbe7 base\n",	1bc6029a-cc97-56da-b8e0-
"pytorch-onnx_rt22.1-py3.9-edt 9d0880bde37f base\n",	1d362186-7ad5-5b59-8b6c-
"tensorflow_2.1-py3.6 3fbdf1665666 base\n",	1eb25b84-d6ed-5dde-b6a5-
"spark-mllib_3.2 a77b012eb8f5 base\n",	20047f72-0a98-58c7-9ff5-
"tensorflow_2.4-py3.8-horovod b19f20564c49 base\n",	217c16f6-178f-56bf-824a-
"runtime-22.1-py3.9-cuda da66306ce658 base\n",	26215f05-08c3-5a41-a1b0-
"do_py3.8 92ae3563e720 base\n",	295addb5-9ef9-547e-9bf4-
"autoai-ts_3.8-py3.8 15e0c2402fb5 base\n",	2aa0c932-798f-5ae9-abd6-
"tensorflow_1.15-py3.6 eae7f436e0bc base\n",	2b73a275-7cbf-420b-a912-
"pytorch_1.2-py3.6 01f94976dac1 base\n",	2c8ef57d-2687-4b7d-acce-
"spark-mllib_2.3 5c6791338875 base\n",	2e51f700-bca0-4b0d-88dc-
"pytorch-onnx_1.1-py3.6-edt dde874a8d67e base\n",	32983cea-3f32-4400-8965-
"spark-mllib_3.0-py37 eafe787600e9 base\n",	36507ebe-8770-55ba-ab2a-
"spark-mllib_2.4 d7ceda621326 base\n",	390d21f8-e58b-4fac-9c55-
"xgboost_0.82-py3.6 60233c80306e base\n",	39e31acd-5f30-41dc-ae44-
"pytorch-onnx_1.2-py3.6-edt fb03b6f4fe12 base\n",	40589d0e-7019-4e28-8daa-
"default_r36py38 8580229facf0 base\n",	41c247d3-45f8-5a71-b065-
"autoai-ts_rt22.1-py3.9 2d495b0c71f7 base\n",	4269d26e-07ba-5d40-8f66-
"autoai-obm_3.0 4240baled5f7 base\n",	42b92e18-d9ab-567f-988a-
"pmml-3.0_4.3 81b8af80e9c7 base\n",	493bcb95-16f1-5bc5-bee8-
"spark-mllib_2.4-r_3.6 a42d0021c095 base\n",	49403dff-92e9-4c87-a3d7-
"xgboost_0.90-py3.6 689c965304d3 base\n",	4ff8d6c2-1343-4c18-85e1-
"pytorch-onnx_1.1-py3.6 b0bed208c60b base\n",	50f95b2a-bc16-43bb-bc94-
"autoai-ts_3.9-py3.8 a5e7cbb42cde base\n",	52c57136-80fa-572e-8728-
"spark-mllib_2.4-scala_2.11 9edb5a443af5 base\n",	55a70f99-7320-4be5-9fb9-
"spark-mllib_3.0 ffd44ea8ffe9 base\n",	5c1b0ca2-4977-5c2e-9439-
"autoai-obm_2.0 d912469614ee base\n",	5c2e37fa-80b8-5e77-840f-
"spss-modeler_18.1 ab53a21dee8b base\n",	5c3cad7e-507f-4b2a-a9a3-
"cuda-py3.8 7bb870a1cd4e base\n",	5d3232bf-c86b-5df4-a2cd-
"autoai-kb_3.1-py3.7 f52dfb6444d7 base\n",	632d4b22-10aa-5180-88f0-

"pytorch-onnx_1.7-py3.8	634d3cdc-b562-5bf9-a2d4-
ea90a478456b base\n",	
"spark-mllib_2.3-r_3.6	6586b9e3-ccd6-4f92-900f-
0f8cb2bd6f0c base\n",	
"tensorflow_2.4-py3.7	65e171d7-72d1-55d9-8ebb-
f813d620c9bb base\n",	
"spss-modeler_18.2	687eddc9-028a-4117-b9dd-
e57b36f1efa5 base\n",	
"pytorch-onnx_1.2-py3.6	692a6a4d-2c4d-45ff-a1ed-
b167ee55469a base\n",	
"spark-mllib_2.3-scala_2.11	7963efe5-bbec-417e-92cf-
0574e21b4e8d base\n",	
"spark-mllib_2.4-py37	7abc992b-b685-532b-a122-
a396a3cdbaab base\n",	
"caffe_1.0-py3.6	7bb3dbe2-da6e-4145-918d-
b6d84aa93b6b base\n",	
"pytorch-onnx_1.7-py3.7	812c6631-42b7-5613-982b-
02098e6c909c base\n",	
"cuda-py3.6	82c79ece-4d12-40e6-8787-
a7b9e0f62770 base\n",	
"tensorflow_1.15-py3.6-horovod	8964680e-d5e4-5bb8-919b-
8342c6c0dfd8 base\n",	
"hybrid_0.1	8c1a58c6-62b5-4dc4-987a-
df751c2756b6 base\n",	
"pytorch-onnx_1.3-py3.7	8d5d8a87-a912-54cf-81ec-
3914adaa988d base\n",	
"caffe-ibm_1.0-py3.6	8d863266-7927-4d1e-97d7-
56a7f4c0a19b base\n",	
"spss-modeler_17.1	902d0051-84bd-4af6-ab6b-
8f6aa6fdeabb base\n",	
"do_12.10	9100fd72-8159-4eb9-8a0b-
a87e12eefa36 base\n",	
"do_py3.7	9447fa8b-2051-4d24-9eef-
5acb0e3c59f8 base\n",	
"spark-mllib_3.0-r_3.6	94bb6052-c837-589d-83f1-
f4142f219e32 base\n",	
"cuda-py3.7-opence	94e9652b-7f2d-59d5-ba5a-
23a414ea488f base\n",	
"nlp-py3.8	96e60351-99d4-5a1c-9cc0-
473ac1b5a864 base\n",	
"cuda-py3.7	9a44990c-1aa1-4c7d-baf8-
c4099011741c base\n",	
"hybrid_0.2	9b3f9040-9cee-4ead-8d7a-
780600f542f7 base\n",	
"spark-mllib_3.0-py38	9f7a8fc1-4d3c-5e65-ab90-
41fa8de2d418 base\n",	
"autoai-kb_3.3-py3.7	a545cca3-02df-5c61-9e88-
998b09dc79af base\n",	
"spark-mllib_3.0-py39	a6082a27-5acc-5163-b02c-
6b96916eb5e0 base\n",	
"runtime-22.1-py3.9-do	a7e7dbf1-1d03-5544-994d-
e5ec845ce99a base\n",	
"default_py3.8	ab9e1b80-f2ce-592c-a7d2-
4f2344f77194 base\n",	
"tensorflow_rt22.1-py3.9	acd9c798-6974-5d2f-a657-
ce06e986df4d base\n",	
"kernel-spark3.2-py3.9	ad7033ee-794e-58cf-812e-
a95f4b64b207 base\n",	
"autoai-obm_2.0 with Spark 3.0	af10f35f-69fa-5d66-9bf5-
acb58434263a base\n",	

"default_py3.7_opence	c2057dd4-f42c-5f77-a02f-
72bdbc3282c9 base\n",	
"tensorflow_2.1-py3.7	c4032338-2a40-500a-beef-
b01ab2667e27 base\n",	
"do_py3.7_opence	cc8f8976-b74a-551a-bb66-
6377f8d865b4 base\n",	
"autoai-kb_3.0-py3.6	d139f196-e04b-5d8b-9140-
9a10ca1fa91a base\n",	
"spark-mllib_3.0-py36	d82546d5-dd78-5fbb-9131-
2ec309bc56ed base\n",	
"autoai-kb_3.4-py3.8	da9b39c3-758c-5a4f-9cfd-
457dd4d8c395 base\n",	
"kernel-spark3.2-r3.6	db2fe4d6-d641-5d05-9972-
73c654c60e0a base\n",	
"autoai-kb_rt22.1-py3.9	db6afe93-665f-5910-b117-
d879897404d9 base\n",	
"tensorflow_rt22.1-py3.9-horovod	dda170cc-ca67-5da7-9b7a-
cf84c6987fae base\n",	
"autoai-ts_1.0-py3.7	deef04f0-0c42-5147-9711-
89f9904299db base\n",	
"tensorflow_2.1-py3.7-horovod	e384fce5-fdd1-53f8-bc71-
11326c9c635f base\n",	
"default_py3.7	e4429883-c883-42b6-87a8-
f419d64088cd base\n",	
"do_22.1	e51999ba-6452-5f1f-8287-
17228b88b652 base\n",	
"autoai-obm_3.2	eae86aab-da30-5229-a6a6-
1d0d4e368983 base\n",	
"do_20.1	f686cdd9-7904-5f9d-a732-
01b0d6b10dc5 base\n",	
"scikit-learn_0.19-py3.6	f963fa9d-4bb7-5652-9c5d-
8d9289ef6ad9 base\n",	
"tensorflow_2.4-py3.8	fe185c44-9a99-5425-986b-
59bd1d2eda46 base\n",	
"-----	-----
- ----\n"	
]	
}	
],	
"source": [
"client.software_specifications.list(limit=100)"	
]	
},	
{	
"cell_type": "code",	
"execution_count": 33,	
"metadata": {},	
"outputs": [
{	
"data": {	
"text/plain": [
"'2.9.1'"	
]	
},	
"execution_count": 33,	
"metadata": {},	
"output_type": "execute_result"	
}	
],	
"source": [
"import tensorflow\n",	

```

        "tensorflow.__version__"
    ]
},
{
    "cell_type": "code",
    "execution_count": 34,
    "metadata": {},
    "outputs": [
        {
            "data": {
                "text/plain": [
                    "'65e171d7-72d1-55d9-8ebb-f813d620c9bb'"
                ]
            },
            "execution_count": 34,
            "metadata": {},
            "output_type": "execute_result"
        }
    ],
    "source": [
        "software_spec_uid =
client.software_specifications.get_uid_by_name(\"tensorflow_2.4-
py3.7\")\n",
        "software_spec_uid"
    ]
},
{
    "cell_type": "code",
    "execution_count": 38,
    "metadata": {},
    "outputs": [],
    "source": [
        "model_details = client.repository.store_model(model='ECG-Image-based-
heartbeat-classification-model_new.tgz',meta_props={\n",
        "client.repository.ModelMetaNames.NAME:\n\"image_classification\",\n",
        "client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:
client.software_specifications.get_uid_by_name(\"tensorflow_2.4-
py3.7\"),\n",
        "client.repository.ModelMetaNames.TYPE:\n\"keras_2.2.5\"})\n",
        "\n",
        "model_id = client.repository.get_model_uid(model_details)\n",
        "    "
    ]
},
{
    "cell_type": "code",
    "execution_count": 41,
    "metadata": {},
    "outputs": [
        {
            "name": "stdout",
            "output_type": "stream",
            "text": [
                "Successfully saved model content to file: 'my_model.tar.gz'\n"
            ]
        }
    ],
    "source": [
        "client.repository.download(model_id, 'my_model.tar.gz')"
    ]
},

```

```

{
  "cell_type": "code",
  "execution_count": 42,
  "metadata": {},
  "outputs": [],
  "source": [
    "from tensorflow.keras.models import load_model\n",
    "from tensorflow.keras.preprocessing import image"
  ]
},
{
  "cell_type": "code",
  "execution_count": 43,
  "metadata": {},
  "outputs": [],
  "source": [
    "model = load_model(\"ECG.h5\")"
  ]
},
],
"metadata": {
  "kernelspec": {
    "display_name": "Python 3.10.2 64-bit",
    "language": "python",
    "name": "python3"
  },
  "language_info": {
    "codemirror_mode": {
      "name": "ipython",
      "version": 3
    },
    "file_extension": ".py",
    "mimetype": "text/x-python",
    "name": "python",
    "nbconvert_exporter": "python",
    "pygments_lexer": "ipython3",
    "version": "3.10.2"
  },
  "vscode": {
    "interpreter": {
      "hash":
"26de051ba29f2982a8de78e945f0abaf191376122a1563185a90213a26c5da77"
    }
  }
},
"nbformat": 4,
"nbformat_minor": 4
}

```