DICOM (& Python🐍)

Digital Imaging and Communications in Medicine

Gal Goldner

PyWeb-IL #109: September 2024

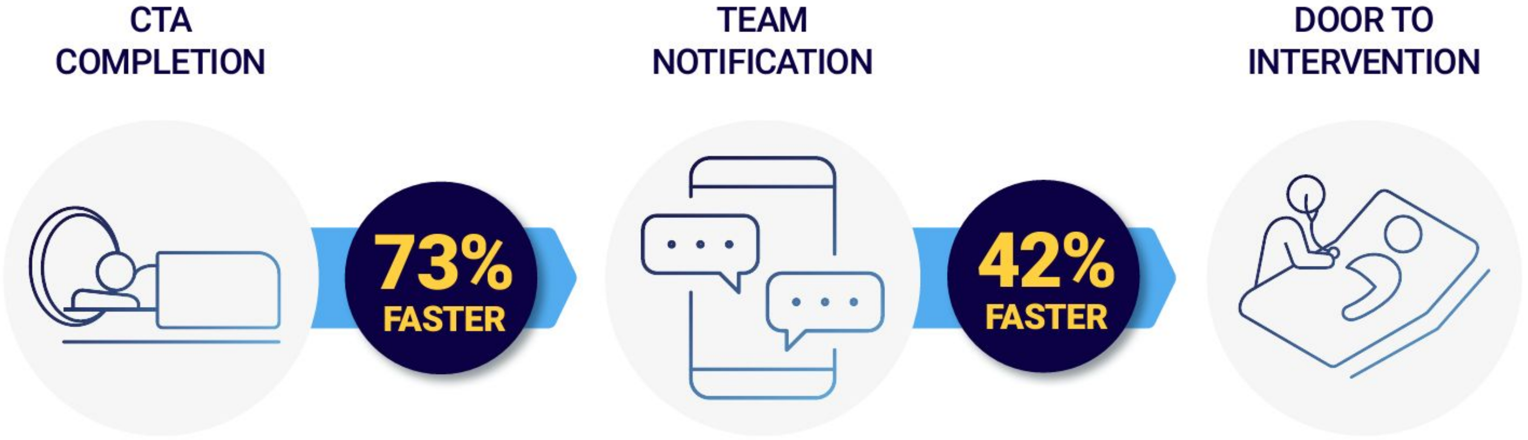
Gal Goldner

Viz.ai


NCCT
CTA
CTP
MIP
3D
Sonia Martinez
74y | F | MRN: 1K34MF095L1
St. Hill Demo
5:02
Thursday, March 2
VIZ.AI SYSTEM
Suspected LVO Identified:
74F @ St Hill Hospital
Your urgent review is requested
Press for more
VIZ.AI SYSTEM
Completed CTP:
74F @ St Hill Hospital
TMax>6s; 98.2cc, CBF<30% 31.8cc
Press for more
This conversation is HIPAA compliant
John Smith
5:13 PM
Neurosurgeon
The scan just came back. Has Dr. Dawan been alerted?
Chrissy James
5:13 PM
Stroke Program Coordinator
Yes, on her way.
Priya Dawan
5:18 PM
Neurosurgeon
Hyperdense right MCA. Consenting now. Chrissy, please confirm that Anesthesia is on the way.
5:47 PM

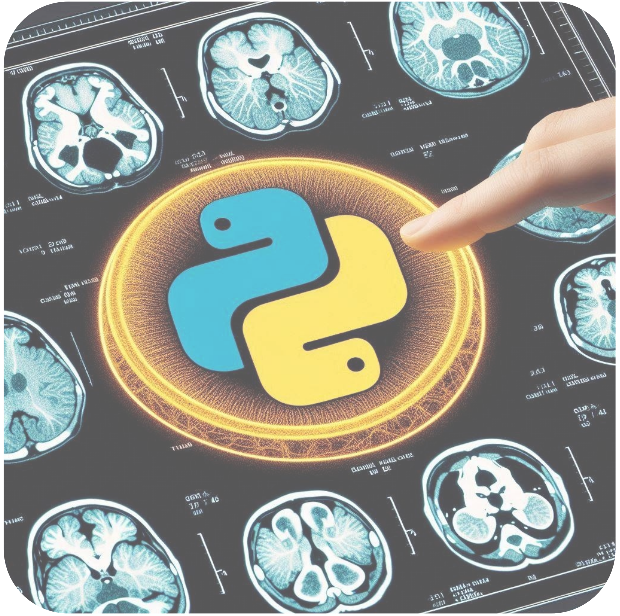
Viz.ai

Significantly save time and improve patient and economic outcomes with our AI-powered care coordination



Agenda

- DICOM standard
- DICOM file format
- pydicom 🐍
- DICOM network protocol
- pynetdicom 🐍



DICOM Standard

- DICOM (Digital Imaging and Communications in Medicine) was developed in the 1980s by the American College of Radiology (ACR) and the National Electrical Manufacturers Association (NEMA) to standardize the handling and sharing of medical images

- Before DICOM, compatibility issues between different devices made medical imaging workflows inefficient

- Since its first release in 1985, DICOM has become the global standard for the storage, transmission, and display of medical images, ensuring seamless interoperability across healthcare systems

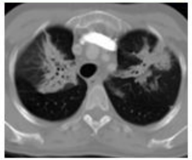
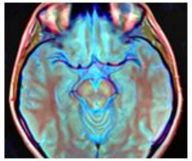
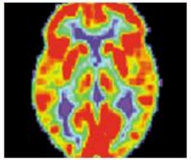
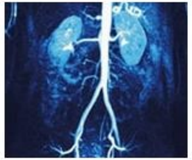
DICOM Standard

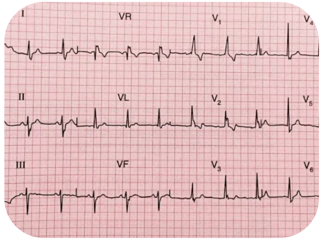
- DICOM (Digital Imaging and Communications in Medicine) is a technical standard for the digital storage and transmission of medical images and related information

- It includes a **file format definition**, which specifies the structure of a DICOM file, as well as a **network communication protocol** that uses TCP/IP to communicate between systems

- The primary purpose of the standard is to facilitate communication between the software and hardware entities involved in medical imaging, especially those that are created by different manufacturers

DICOM Standard

	X-ray	CT (CAT) scan	MRI	Ultrasound	PET scanning	Nuclear scanning
						
	Radiography	Computed (Axial) Tomography	Magnetic Resonance Imaging	Ultra-sonography	Positron Emission Tomography	Radionuclide Imaging
Major uses	Bons, skull, abdomen, chest, & lungs.	Chest & area between the lungs, upper abdomen & peritoneal cavity (surrounds the organs of the abdomen). Brain & spine.	Brain, spinal cord, coronary arteries, heart. Organs not in constant motion. Soft tissues, liver, bile ducts, kidneys, spleen. Ligaments & cartilage in large joints, knee, hip, shoulder, & jaw.	Reproductive organs. Uniform & solid, constant motion, or fluid-filled organs. Soft tissue. Heart, gallbladder, liver, abdominal, urinary tract, & thyroid gland.	Track biochemical changes, chemical functions & processes in body tissues & organs - blood flow & metabolism. Visualize any body region.	Bons, heart, lungs, kidneys, bladders, thyroid, & gallbladder.



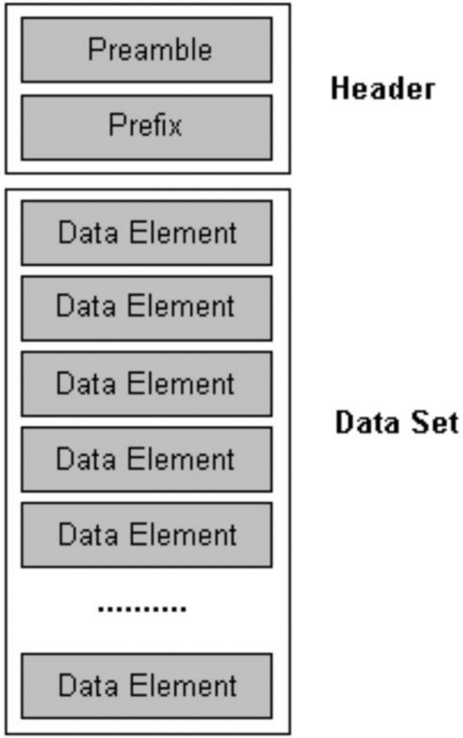
DICOM File format

DICOM file is comprised of a **Header** and a **Data Set**:

- **Header**, also known as DICOM File Meta Information, includes a preamble, followed by 128 byte File Preamble, followed by 4 byte DICOM prefix (‘D’, ‘I’, ‘C’, ‘M’), followed by the File Meta Elements which include elements such as the TransferSyntaxUID (which is important for understanding the file format).

- **Data Set** is a collection of Data Elements.

DICOM File format



DICOM basic file structure

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000080	44	49	43	4D	02	00	00	00	55	4C	04	00	D0	00	00	00	DICM....UL..Ð...
00000090	02	00	01	00	4F	42	00	00	02	00	00	00	00	01	02	00	....OB..........
000000A0	02	00	55	49	1A	00	31	2E	32	2E	38	34	30	2E	31	30	..UI..1.2.840.10
000000B0	30	30	38	2E	35	2E	31	2E	34	2E	31	2E	31	2E	32	00	008.5.1.4.1.1.2.
000000C0	02	00	03	00	55	49	3E	00	31	2E	33	2E	36	2E	31	2E	....UI>.1.3.6.1.
000000D0	34	2E	31	2E	39	35	39	30	2E	31	30	30	2E	31	2E	32	4.1.9590.100.1.2
000000E0	2E	35	33	33	31	35	30	36	39	31	32	32	33	31	35	37	.533150691223157
000000F0	34	30	30	39	31	33	37	30	35	33	35	32	30	31	35	30	4009137053520150
00000100	35	38	38	34	33	38	02	00	10	00	55	49	14	00	31	2E	588438....UI..1.
00000110	32	2E	38	34	30	2E	31	30	30	30	38	2E	31	2E	32	2E	2.840.10008.1.2.
00000120	31	00	02	00	12	00	55	49	20	00	31	2E	33	2E	36	2E	1.....UI .1.3.6.
00000130	31	2E	34	2E	31	2E	39	35	39	30	2E	31	30	30	2E	31	1.4.1.9590.100.1
00000140	2E	33	2E	31	30	30	2E	39	2E	34	02	00	13	00	53	48	.3.100.9.4....SH
00000150	0E	00	4D	41	54	4C	41	42	20	49	50	54	20	39	2E	34	..MATLAB IPT 9.4
00000160	08	00	05	00	43	53	0A	00	49	53	4F	5F	49	52	20	31	....CS..ISO_IR 1
00000170	30	30	08	00	08	00	43	53	1E	00	4F	52	49	47	49	4E	00....CS..ORIGIN
00000180	41	4C	5C	50	52	49	4D	41	52	59	5C	41	58	49	41	4C	AL\PRIMARY\AXIAL
00000190	5C	48	45	4C	49	43	41	4C	08	00	12	00	44	41	02	00	\HELICAL....DA..
000001A0	30	00	08	00	13	00	54	4D	0A	00	31	37	34	34	34	34	0.....TM..174444
000001B0	2E	38	37	35	08	00	16	00	55	49	1A	00	31	2E	32	2E	.875....UI..1.2.
000001C0	38	34	30	2E	31	30	30	30	38	2E	35	2E	31	2E	34	2E	840.10008.5.1.4.
000001D0	31	2E	31	2E	32	00	08	00	18	00	55	49	3E	00	31	2E	1.1.2.....UI>.1.
000001E0	33	2E	36	2E	31	2E	34	2E	31	2E	39	35	39	30	2E	31	3.6.1.4.1.9590.1
000001F0	30	30	2E	31	2E	32	2E	35	33	33	31	35	30	36	39	31	00.1.2.533150691
00000200	32	32	33	31	35	37	34	30	30	39	31	33	37	30	35	33	2231574009137053
00000210	35	32	30	31	35	30	35	38	38	34	33	38	08	00	20	00	520150588438.. .
00000220	44	41	02	00	30	00	08	00	22	00	44	41	02	00	30	00	DA..0...".DA..0.
00000230	08	00	23	00	44	41	02	00	30	00	08	00	30	00	54	4D	..#.DA..0...0.TM
00000240	02	00	30	20	08	00	32	00	54	4D	02	00	30	20	08	00	..0 ..2.TM..0 ..
00000250	33	00	54	4D	02	00	30	20	08	00	50	00	53	48	02	00	3.TM..0 ..P.SH..
00000260	30	20	08	00	60	00	43	53	02	00	43	54	08	00	70	00	0 ..`.CS..CT..p.

Hexadecimal representation of DICOM file. Starts with 128 byte File Preamble, followed by a 4 byte DICOM prefix, followed by the File Data Elements

DICOM File format

- Each data element is a unit for storing information and it has a well predefined tag and purpose defined in the DICOM Standard

- Every data element has a Tag that uniquely identifies the element and is represented as: **(gggg,eeee)**, where **gggg represents the Group Number** and **eeee the Element Number**

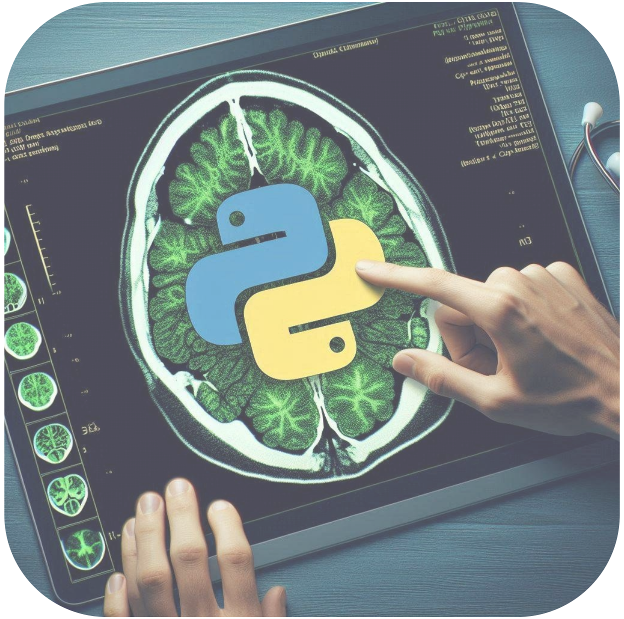
- Many DICOM contain bulk **pixel data**, which is usually used to represent one or more image frames (although other types of data are possible)

Tag	Tag Description
(0010,0010)	Patient’s Name
(0010,0020)	Patient ID
(0010,0021)	Issuer of Patient ID
(0010,0022)	Type of Patient ID
(0010,0030)	Patient’s Birth Date
(0010,0040)	Patient’s Sex
…	…

pydicom

- pydicom is a pure Python package for working with DICOM files such as medical images, reports, and radiotherapy objects

- pydicom makes it easy to read these complex files into natural pythonic structures for easy manipulation. Modified datasets can be written again to DICOM format files



pydicom

```python
import pydicom

ds = pydicom.dcmread("/content/sample_data/cta/21c9e45c2e5a388601897e3a989b1c40be09e4d9/1377938d7fdbceeca328deda548f10d41669670c")
ds
```

```
Dataset.file_meta -------------------------------
(0002, 0000) File Meta Information Group Length  UL: 196
(0002, 0001) File Meta Information Version       OB: b'\x01\x00'
(0002, 0002) Media Storage SOP Class UID         UI: CT Image Storage
(0002, 0003) Media Storage SOP Instance UID      UI: 1.2.840.113619.2.340.3.1074409718.546.1487334849.850.327
(0002, 0010) Transfer Syntax UID                 UI: Explicit VR Little Endian
(0002, 0012) Implementation Class UID            UI: 1.3.46.670589.42.1.4.4.5
(0002, 0013) Implementation Version Name         SH: 'PhilipsISPACS445'
-------------------------------------------------
(0008, 0000) Group Length                        UL: 958
(0008, 0005) Specific Character Set              CS: 'ISO_IR 100'
(0008, 0008) Image Type                          CS: ['ORIGINAL', 'PRIMARY', 'AXIAL']
(0008, 0012) Instance Creation Date              DA: '20170217'
(0008, 0013) Instance Creation Time              TM: '153610'
(0008, 0016) SOP Class UID                       UI: CT Image Storage
(0008, 0018) SOP Instance UID                    UI: 1377938d7fdbceeca328deda548f10d41669670c
(0008, 0020) Study Date                          DA: '20170217'
(0008, 0021) Series Date                         DA: '20170217'
(0008, 0022) Acquisition Date                    DA: '20170217'
(0008, 0023) Content Date                        DA: '20170217'
(0008, 0030) Study Time                          TM: '153358'
(0008, 0031) Series Time                         TM: '153458'
(0008, 0032) Acquisition Time                    TM: '153548.595839'
(0008, 0033) Content Time                        TM: '153610'
(0008, 0050) Accession Number                    SH: 'B7048426'
(0008, 0060) Modality                            CS: 'CT'
(0008, 0070) Manufacturer                        LO: 'GE MEDICAL SYSTEMS'
(0008, 0080) Institution Name                    LO: '0'
(0008, 0081) Institution Address                 ST: '0'
(0008, 0090) Referring Physician's Name          PN: '0'
(0008, 1010) Station Name                        SH: '0'
(0008, 1030) Study Description                   LO: 'CT ANGIOGRAM HEAD NECK W WO CONTRAST'
(0008, 103e) Series Description                  LO: 'AX CTA HEAD NECK THIN'
(0008, 1050) Performing Physician's Name         PN: '0'
(0008, 1060) Name of Physician(s) Reading Study  PN: '0'
(0008, 1070) Operators' Name                     PN: '0'
(0008, 1090) Manufacturer's Model Name           LO: 'Discovery CT750 HD'
```

pydicom

```
len(ds)
```

```
239
```

```
ds.PatientName
```

```
'Moses'
```

```
ds[0x10,0x10].value
```

```
'Moses'
```

```
ds.Modality
```

```
'CT'
```

```
ds.Manufacturer
```

```
'GE MEDICAL SYSTEMS'
```

```
ds.ManufacturerModelName
```

```
'Discovery CT750 HD'
```

pydicom

- By default pydicom reads in pixel data as the raw bytes found in the file:

```
ds.PixelData
```

```
b'\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xf
\x00 \x00\x16\x00\x14\x00\x1e\x00\'\x00*\x00-\x00.\x00.\x00#\x00\x18\x00 \x00.\x00,\x00!\x00\x19\x00\x1d\x00&\x00\x1c\x00\x16\x00
\x00*\x00\'\x00\x13\x00\x18\x008\x002\x00\x15\x00\x17\x00!\x00\x1d\x00\x18\x00\x12\x00\x13\x00\x13\x00\x17\x00\x1d\x00\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x0
\x00\x0f\x00\x1e\x000\x00\x19\x00\n\x00+\x00@\x006\x00+\x00!\x00\x1b\x00,\x006\x006\x005\x00,\x00$\x00\x1f\x00\x18\x00\x1e\x00.\x001\x002\x001\x00(\x00\x1a\x00\x15\x00#\x00/\x00
\x00\x19\x00\x14\x00\x14\x00\x19\x00
\x00\x1f\x00\x0c\x00\x02\x00\x0e\x00\x16\x00\x18\x00\x18\x00\x16\x00\x16\x00\x11\x00\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\
\x00$\x00\x12\x00 \x00)\x00\x11\x00\x05\x00*\x00G\x00=\x00/\x00$\x00\x1b\x00+\x00;\x007\x000\x00.\x00+\x00(\x00
\x00\x1b\x00,\x007\x001\x00.\x00*\x00\x16\x00\x12\x00\'\x002\x000\x00,\x00"\x00\x1f\x00(\x00(\x00#\x00%\x00\x1d\x00\x0f\x00\x03\x00\x15\x008\x000\x00\x12\x00\x14\x00\x1f\x00\x1a
\x00\x1d\x00(\x000\x00,\x00%\x00\x19\x00\x1d\x009\x00@\x00+\x00"\x00\'\x00+\x00\x1a\x00\x0c\x00\'\x00A\x00;\x001\x00&\x00\x1c\x00(\x006\x00.\x00"\x00+\x001\x00+\x00
\x00\x19\x00$\x002\x000\x00/\x00+\x00\x18\x00\x16\x00+\x009\x00=\x009\x00$\x00\x1a\x00%\x00*\x00\'\x00\'\x00\x1e\x00\x11\x00\t\x00"\x00@\x009\x00\x1c\x00\x11\x00\x19\x00\x1a\x00
\x00\x1a\x00\x12\x00\x0c\x00\x0f\x00\x18\x00"\x00&\x00\x18\x00\t\x00\x0c\x00\x11\x00\x17\x00\x1e\x00\x19\x00\r\x00\x06\x00\x12\x00$\x00 \x00\x1b\x00
\x00\x1c\x00\x14\x00\x10\x00\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\xfc\x00\
```

- Because of the complexity in interpreting the pixel data, pydicom provides an easy way to get it in a convenient form: Dataset.pixel_array

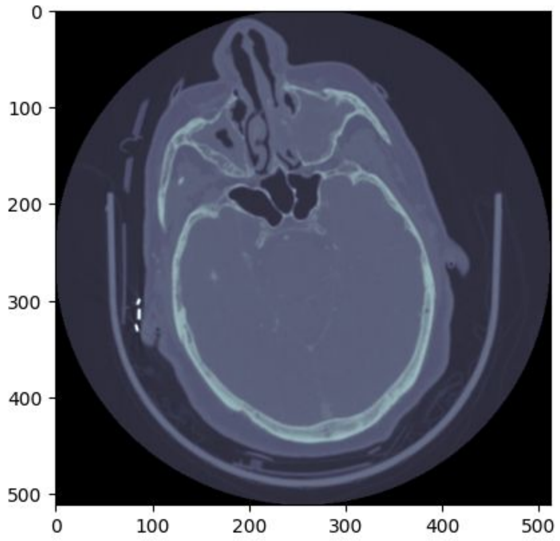
```
ds.pixel_array
```

```
array([[-1024, -1024, -1024, ..., -1024, -1024, -1024],
       [-1024, -1024, -1024, ..., -1024, -1024, -1024],
       [-1024, -1024, -1024, ..., -1024, -1024, -1024],
       ...,
       [-1024, -1024, -1024, ..., -1024, -1024, -1024],
       [-1024, -1024, -1024, ..., -1024, -1024, -1024],
       [-1024, -1024, -1024, ..., -1024, -1024, -1024]], dtype=int16)
```

pydicom

```python
import pydicom
import matplotlib.pyplot as plt

ds = pydicom.dcmread("/content/sample_data/cta/21c9e45c2e5a388601897e3a989b1c40be09e4d9/6b7648c546f2b24f8eff8939e86b36c8e210e448")
plt.imshow(ds.pixel_array)
```

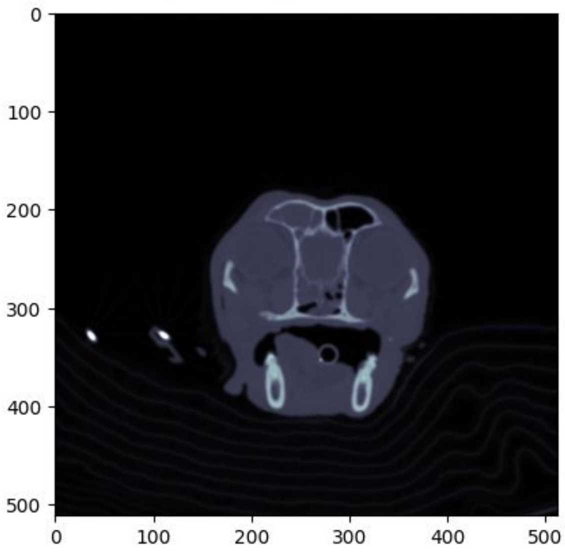
```
<matplotlib.image.AxesImage at 0x7ac7e81801c0>
```



pydicom

```
import pydicom
import matplotlib.pyplot as plt

ds = pydicom.dcmread("/content/sample_data/ct/series-00001/image-00180.dcm")
plt.imshow(ds.pixel_array)
```

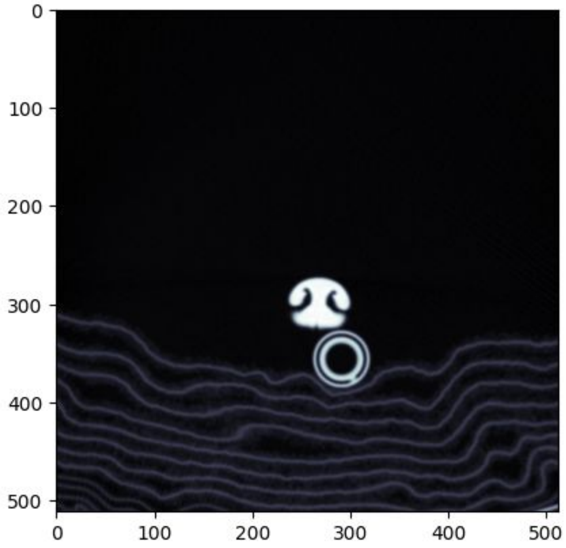
```
<matplotlib.image.AxesImage at 0x7ac7e7cc2440>
```



pydicom

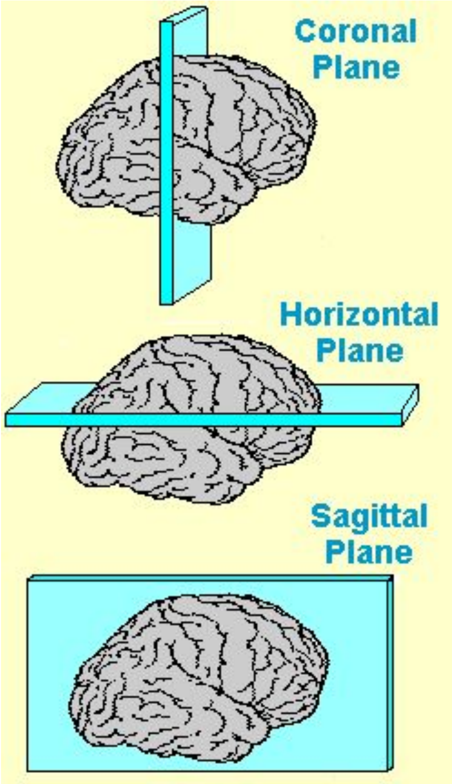
```
import pydicom
import matplotlib.pyplot as plt

ds = pydicom.dcmread("/content/sample_data/ct/series-00001/image-00019.dcm")
plt.imshow(ds.pixel_array)
```

```
<matplotlib.image.AxesImage at 0x7ac7e813b3a0>
```



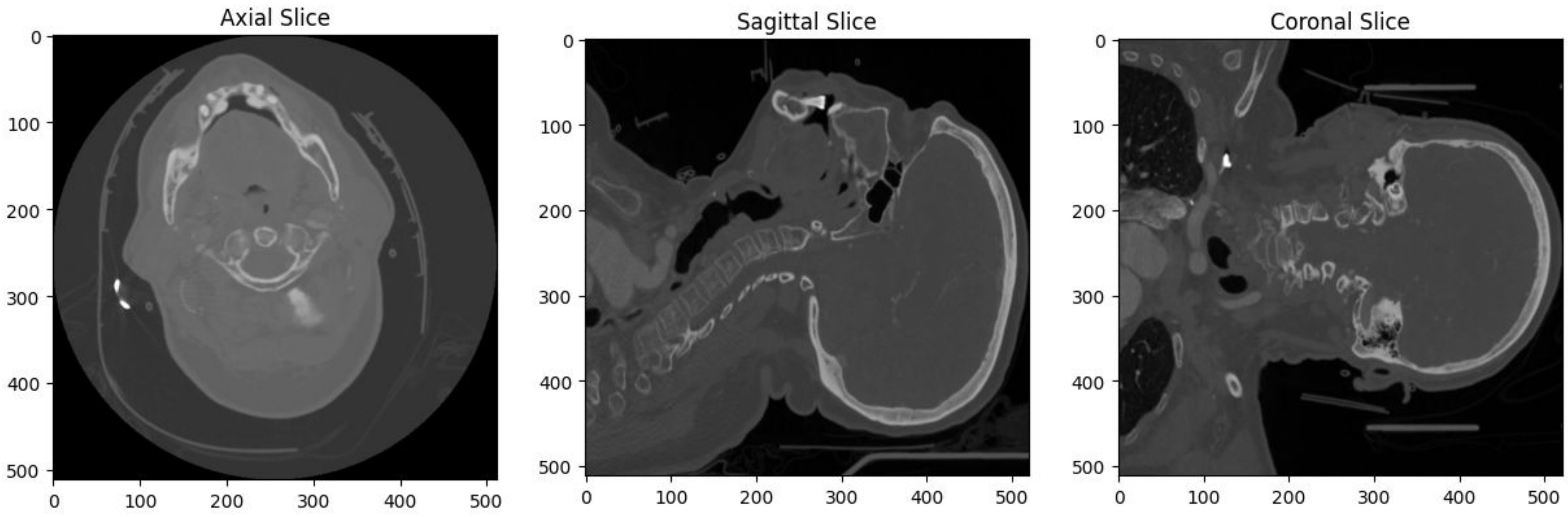
pydicom

```python
# List all files in the directory
directory = "/content/sample_data/ct/series-00001"
dicom_files = [os.path.join(directory, f) for f in os.listdir(directory)]

# Read all DICOM files
slices = [pydicom.dcmread(f) for f in dicom_files]

# Sort slices based on Instance Number (to get the correct order)
slices.sort(key=lambda x: int(x.InstanceNumber))

# Stack slices to create a 3D volume
scan = np.stack([s.pixel_array for s in slices])

# Plotting the slices
plt.figure(figsize=(15, 5))

# Axial slice (XY plane)
mid_axial = scan.shape[2] // 2
axial_slice = scan[:, :, mid_axial]
plt.subplot(1, 3, 1)
plt.imshow(axial_slice, cmap='gray')
plt.title('Axial Slice')

# Coronal slice  (XZ plane)
mid_coronal = scan.shape[1] // 2
coronal_slice = scan[:, mid_coronal, :]
plt.subplot(1, 3, 2)
plt.imshow(coronal_slice, cmap='gray')
plt.title('Coronal Slice')

# Sagittal slice (YZ plane)
mid_sagittal = scan.shape[0] // 2
sagittal_slice = scan[mid_sagittal, :, :]
plt.subplot(1, 3, 3)
plt.imshow(sagittal_slice, cmap='gray')
plt.title('Sagittal Slice')
```



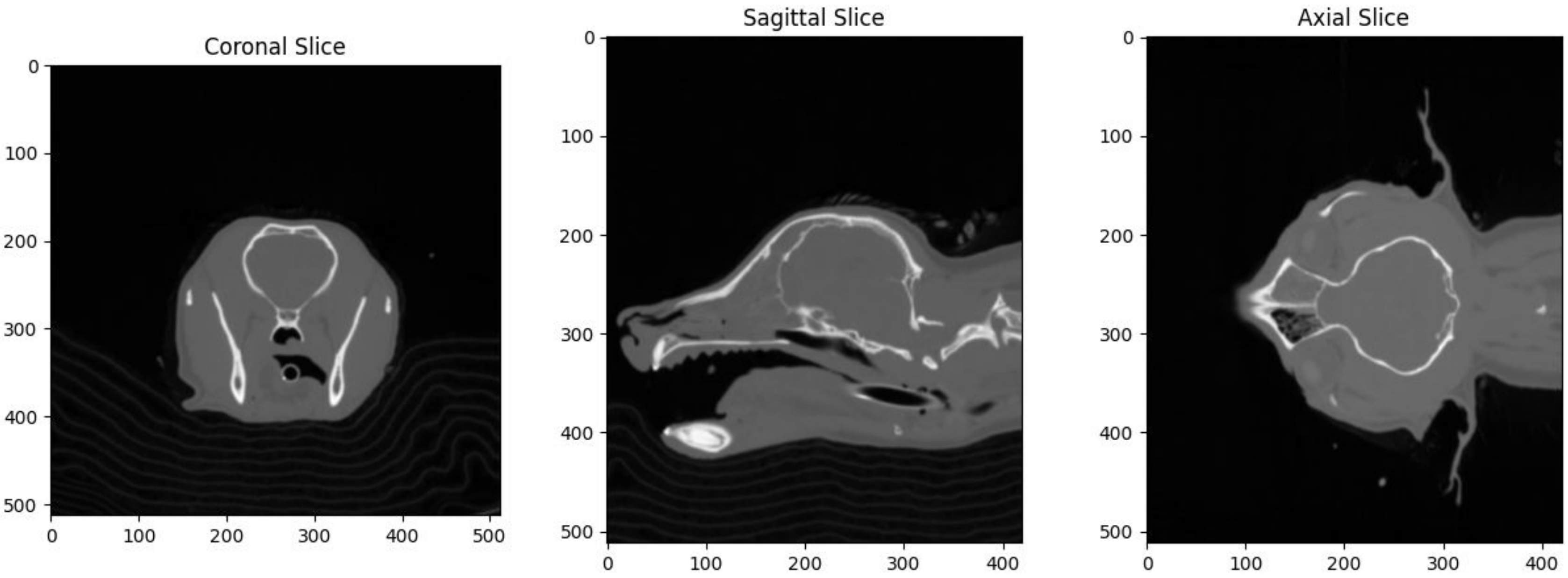
pydicom

```
plt.show()
```



pydicom

```
plt.show()
```

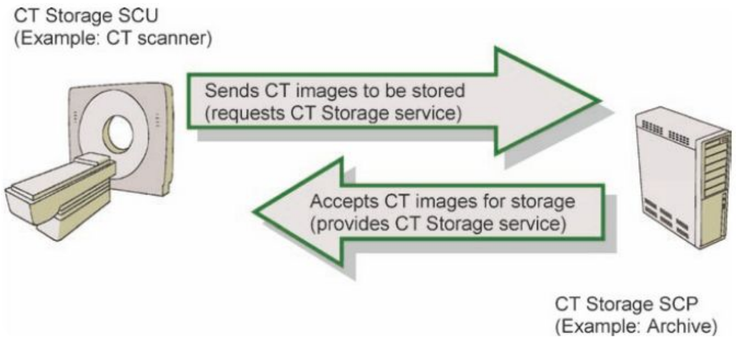


DICOM network protocol

- DICOM is a widely adopted network protocol used primarily in medical imaging to ensure the seamless exchange of information between imaging devices, such as X-rays, MRIs, and CT scanners, and healthcare systems like PACS (Picture Archiving and Communication Systems)

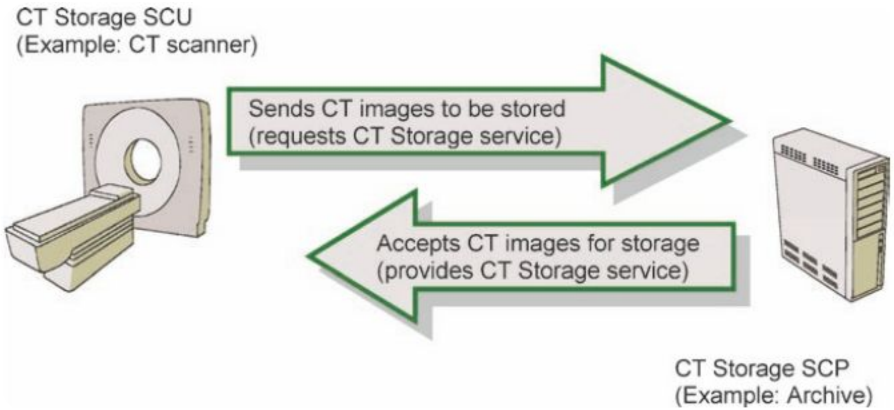
- It defines a standardized format for storing, transmitting, and sharing medical images and associated data, enabling interoperability across various equipment from different manufacturers

- **DICOM uses a client-server model and supports communication over TCP/IP**



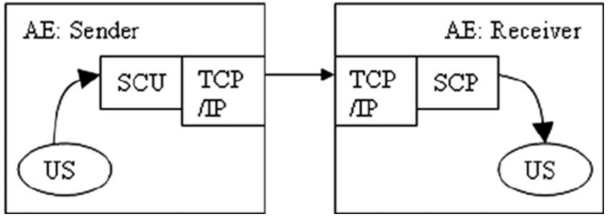
pynetdicom

- pynetdicom is a pure Python package that implements the DICOM networking protocol
- Working with pydicom, it allows DICOM clients (SCUs - Service Class Users) and servers (SCPs - Service Class Providers) to be easily created



pynetdicom Application Entity + Association

pynetdicom's main user class is AE and is used to represent a **DICOM Application Entity**



With it you can:

- Start the application as an SCP by specifying the supported presentation contexts then calling AE.start_server() and **waiting for incoming association requests**
- Use the application as an SCU by specifying the presentation contexts you want the peer SCP to support, then **requesting an association** via the AE.associate() method, which returns an Association thread

Once associated, the services available to the association can be used by **sending DIMSE messages**

(DIMSE = DICOM Message Service Element)

pynetdicom DICOM Message Service Element

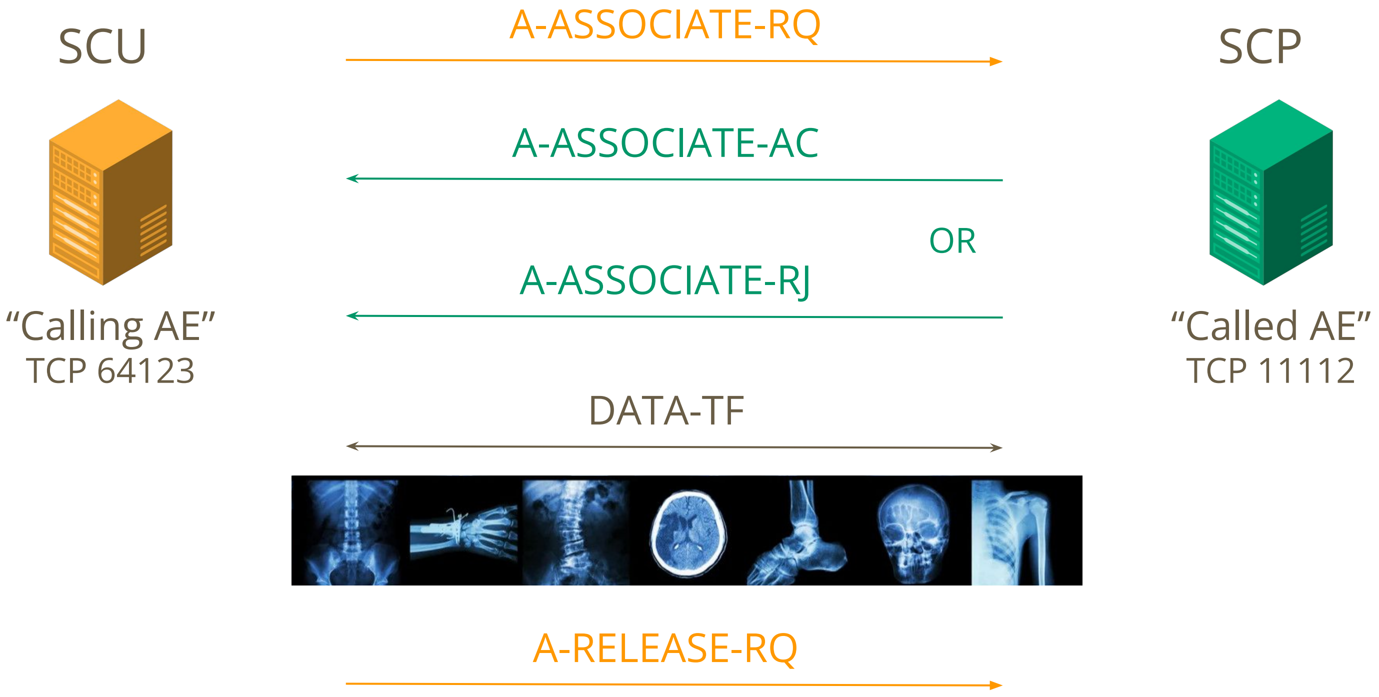
DICOM Messages are used to communicate information across the DICOM network

A DICOM Message is a DICOM Command Set, and can be followed by a conditional DICOM Data Set

The Command Set indicates the operations and/or notifications that the requesting Application Entity (AE) wishes to have the accepting AE perform on or with the conditional Data Set

Name	Group	Type
C-STORE	DIMSE-C	operation
C-GET	DIMSE-C	operation
C-MOVE	DIMSE-C	operation
C-FIND	DIMSE-C	operation
C-ECHO	DIMSE-C	operation

pynetdicom DICOM Network Protocol



pynetdicom DICOM Network Protocol

scp.py

```python
from pydicom.uid import ExplicitVRLittleEndian

from pynetdicom import AE, debug_logger, evt
from pynetdicom.sop_class import CTImageStorage
from pynetdicom.status import Status

debug_logger()

print("SCP is listening for incoming DICOMs ⬇️")

ae = AE()
ae.add_supported_context(CTImageStorage, ExplicitVRLittleEndian)
ae.start_server(("127.0.0.1", 11112))
```

What?
CT, MRI, US, etc

How?
ILE, ELE, JPEG, etc

SCP

"Called AE"
TCP 11112

DICOM Receiver

SCP Services

When the AE is acting as an SCP the following DIMSE-C and -N services are available to the peer once an association has been established:

DIMSE service	Intervention Event	Handler documentation
C-ECHO	`evt.EVT_C_ECHO`	Handle C-ECHO
C-FIND	`evt.EVT_C_FIND`	Handle C-FIND
C-GET	`evt.EVT_C_GET`	Handle C-GET
C-MOVE	`evt.EVT_C_MOVE`	Handle C-MOVE
C-STORE	`evt.EVT_C_STORE`	Handle C-STORE
N-ACTION	`evt.EVT_N_ACTION`	Handle N-ACTION
N-CREATE	`evt.EVT_N_CREATE`	Handle N-CREATE
N-DELETE	`evt.EVT_N_DELETE`	Handle N-DELETE
N-EVENT-REPORT	`evt.EVT_N_EVENT_REPORT`	Handle N-EVENT-REPORT
N-GET	`evt.EVT_N_GET`	Handle N-GET
N-SET	`evt.EVT_N_SET`	Handle N-SET

pynetdicom DICOM Network Protocol

scp.py

```python
import uuid

from pydicom.uid import ExplicitVRLittleEndian
from pynetdicom import AE, debug_logger, evt
from pynetdicom.sop_class import CTImageStorage
from pynetdicom.status import Status


debug_logger()


def handle_store(event):
    """Handle EVT_C_STORE events"""
    ds = event.dataset
    ds.file_meta = event.file_meta
    ds.save_as(f"/Users/gal.goldner/Downloads/{uuid.uuid4()}.dcm")
    print(f"📝 Patient name: {ds.PatientName}")
    return Status.SUCCESS


print("SCP is listening for incoming DICOMs ⬇️")

handlers = [(evt.EVT_C_STORE, handle_store)]

ae = AE()
ae.add_supported_context(CTImageStorage, ExplicitVRLittleEndian)
ae.start_server(("127.0.0.1", 11112), evt_handlers=handlers)
```

```
SCP is listening for incoming DICOMs ⬇️
D: Request Parameters:
D: ====================== INCOMING A-ASSOCIATE-RQ PDU ======================
D: Their Implementation Class UID:      1.2.826.0.1.3680043.9.3811.2.1.0
D: Their Implementation Version Name:   PYNETDICOM_210
D: Application Context Name:    1.2.840.10008.3.1.1.1
D: Calling Application Name:    STORESCU
D: Called Application Name:     ANY-SCP
D: Their Max PDU Receive Size:  16382
D: Presentation Context:
D:   Context ID:        1 (Proposed)
D:     Abstract Syntax: =CT Image Storage
D:     Proposed SCP/SCU Role: Default
D:     Proposed Transfer Syntax:
D:       =Explicit VR Little Endian
D: Requested Extended Negotiation: None
D: Requested Common Extended Negotiation: None
D: Requested Asynchronous Operations Window Negotiation: None
D: Requested User Identity Negotiation: None
D: ========================== END A-ASSOCIATE-RQ PDU ==========================
I: Accepting Association
D: Accept Parameters:
D: ====================== OUTGOING A-ASSOCIATE-AC PDU ======================
D: Our Implementation Class UID:      1.2.826.0.1.3680043.9.3811.2.1.0
D: Our Implementation Version Name:   PYNETDICOM_210
D: Application Context Name:    1.2.840.10008.3.1.1.1
D: Responding Application Name: resp. AE Title
D: Our Max PDU Receive Size:    16382
D: Presentation Contexts:
D:   Context ID:        1 (Accepted)
D:     Abstract Syntax: =CT Image Storage
D:     Accepted SCP/SCU Role: Default
D:     Accepted Transfer Syntax: =Explicit VR Little Endian
D: Accepted Extended Negotiation: None
D: Accepted Asynchronous Operations Window Negotiation: None
D: User Identity Negotiation Response: None
D: ========================== END A-ASSOCIATE-AC PDU ==========================
D: pydicom.read_dataset() TransferSyntax="Little Endian Implicit"
I: Received Store Request
D: ========================== INCOMING DIMSE MESSAGE ==========================
D: Message Type                  : C-STORE RQ
D: Presentation Context ID       : 1
D: Message ID                    : 1
D: Affected SOP Class UID        : CT Image Storage
D: Affected SOP Instance UID     : 1.3.6.1.4.1.5962.1.1.1.1.1.20040119072730.12322
D: Data Set                      : Present
D: Priority                      : Low
D: ============================ END DIMSE MESSAGE ============================
D: pydicom.read_dataset() TransferSyntax="Little Endian Explicit"
📝 Patient name: CITIZEN^John
I: Association Released
```

Thank you!

Gal Goldner