

# Cloud-Based Platform for COPDGene CT Images



Kun-Hsing Yu, MD, PhD

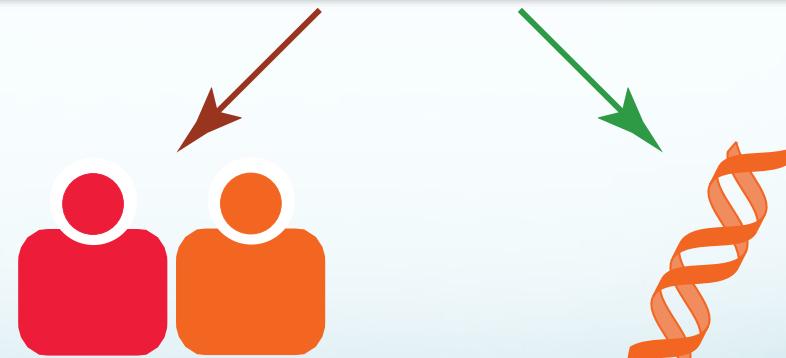
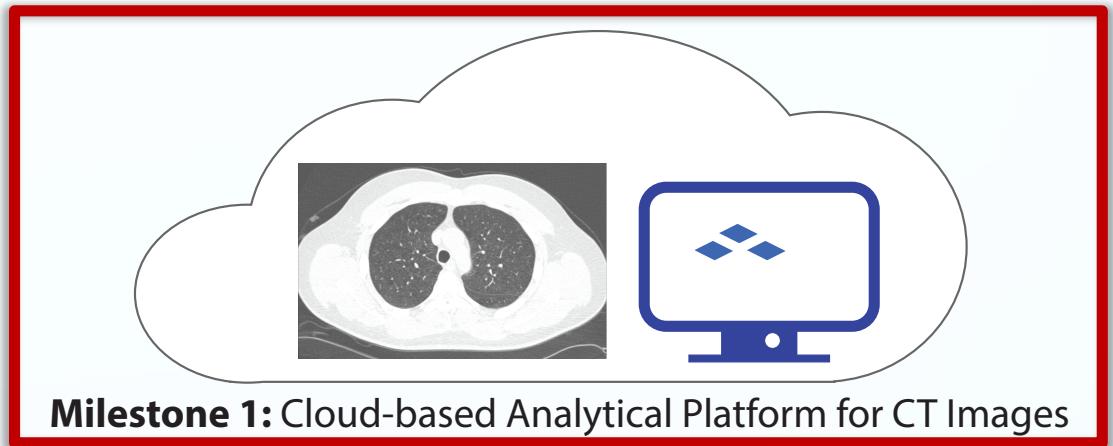


Department of Biomedical Informatics  
Harvard Medical School

July 9<sup>th</sup>, 2018

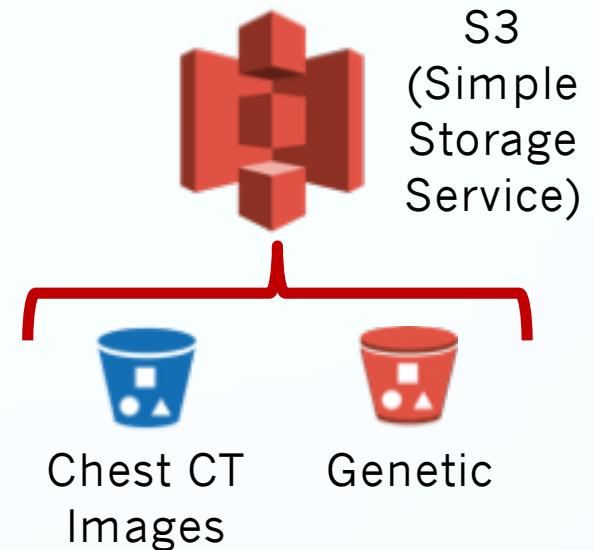
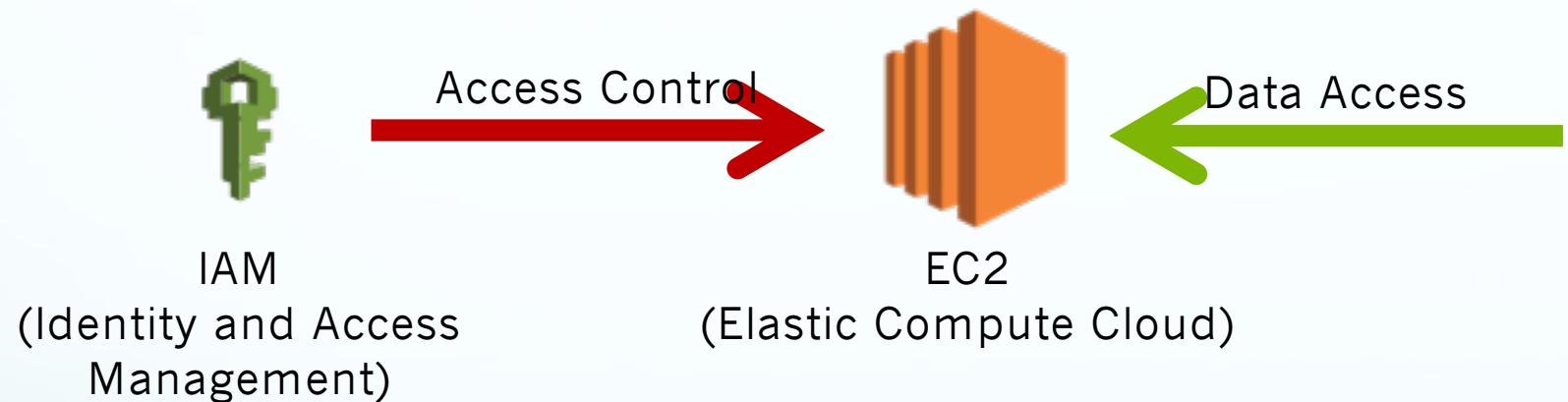
# Outline

- A secure cloud-based analytical platform for
  - Data management
  - Programming interface
  - Visualization
  - 2D and 3D convolutional neural networks analyses





# Platform

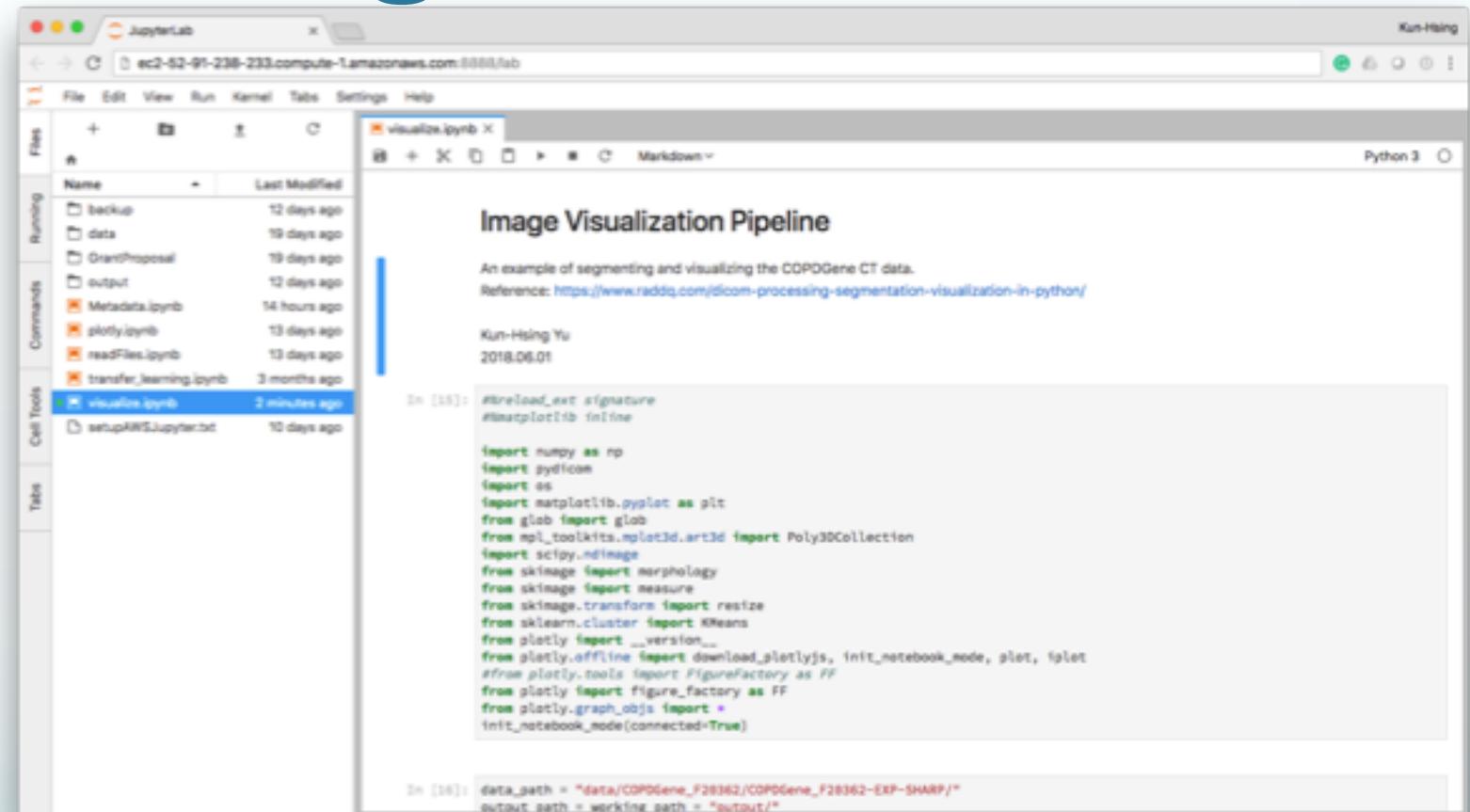


Analytical pipeline also validated on:



# Programming Interface

- Scalable cloud computing
  - Tested on 4-GPU, 8-GPU, and 16-GPU virtual servers



The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** JupyterLab, URL: ec2-52-91-238-233.compute-1.amazonaws.com:8888/lab, User: Kun-Hsing Yu.
- File Menu:** File, Edit, View, Run, Kernel, Tabs, Settings, Help.
- File List:** Shows files like backup, data, GrantProposal, output, Metadata.ipynb, plotly.ipynb, readFiles.ipynb, transferLearning.ipynb, and visualize.ipynb (selected).
- Cell Tools:** Cell, Tools.
- Code Cells:**
  - In [15]:

```
import tensorflow as tf
import numpy as np
import pydicom
import os
import matplotlib.pyplot as plt
from glob import glob
from mpl_toolkits.mplot3d.art3d import Poly3DCollection
import scipy.ndimage
from skimage import morphology
from skimage import measure
from skimage.transform import resize
from sklearn.cluster import KMeans
from plotty import __version__
from plotty import download_plottyjs, init_notebook_mode, plot, spplot
from plotty.tools import FigureFactory as FF
from plotty import figure_factory as FF
from plotty.graph_objs import *
init_notebook_mode(connected=True)
```
  - In [16]:

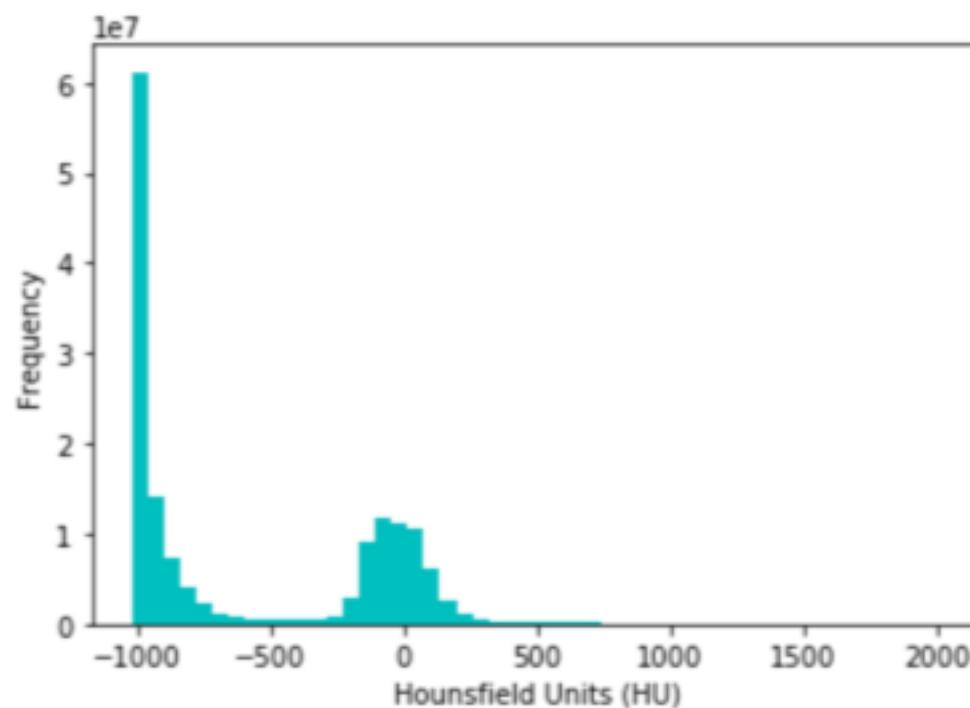
```
data_path = "data/COPDGene_F28362/COPDGene_F28362-EXP-SHARP/"
output_path = "working_path = "output/".
```

- Prototype: <http://rebrand.ly/copdgene> (password protected)

# Data & Metadata Access

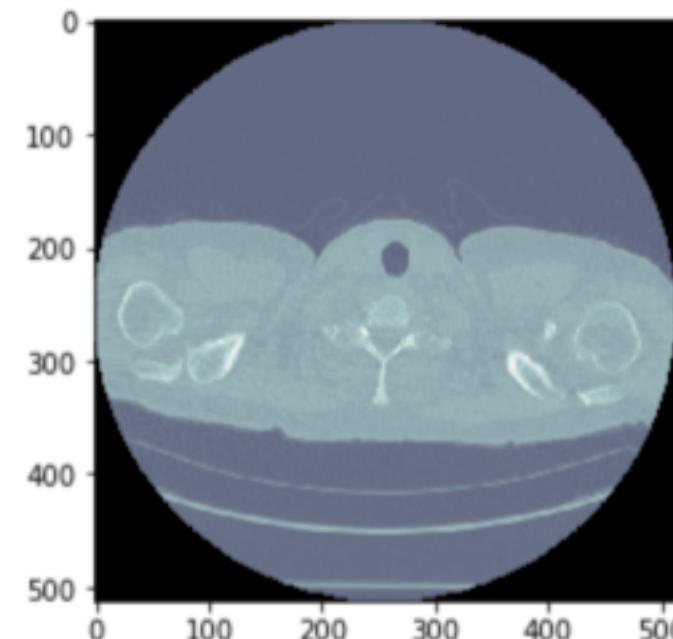
## Metadata

```
In [3]: print("Filename.....:", filename)
print("Storage type.....:", dataset.SOPClassUID)
print("Patient id.....:", dataset.PatientID)
print("Modality.....:", dataset.Modality)
print("Study Date.....:", dataset.StudyDate)
if 'PixelData' in dataset:
    rows = int(dataset.Rows)
    cols = int(dataset.Columns)
    print ("Image size.....: {rows:d} x {cols:d}, {size:d} bytes".format(
        rows=rows, cols=cols, size=dataset.Rows * dataset.Columns))
```

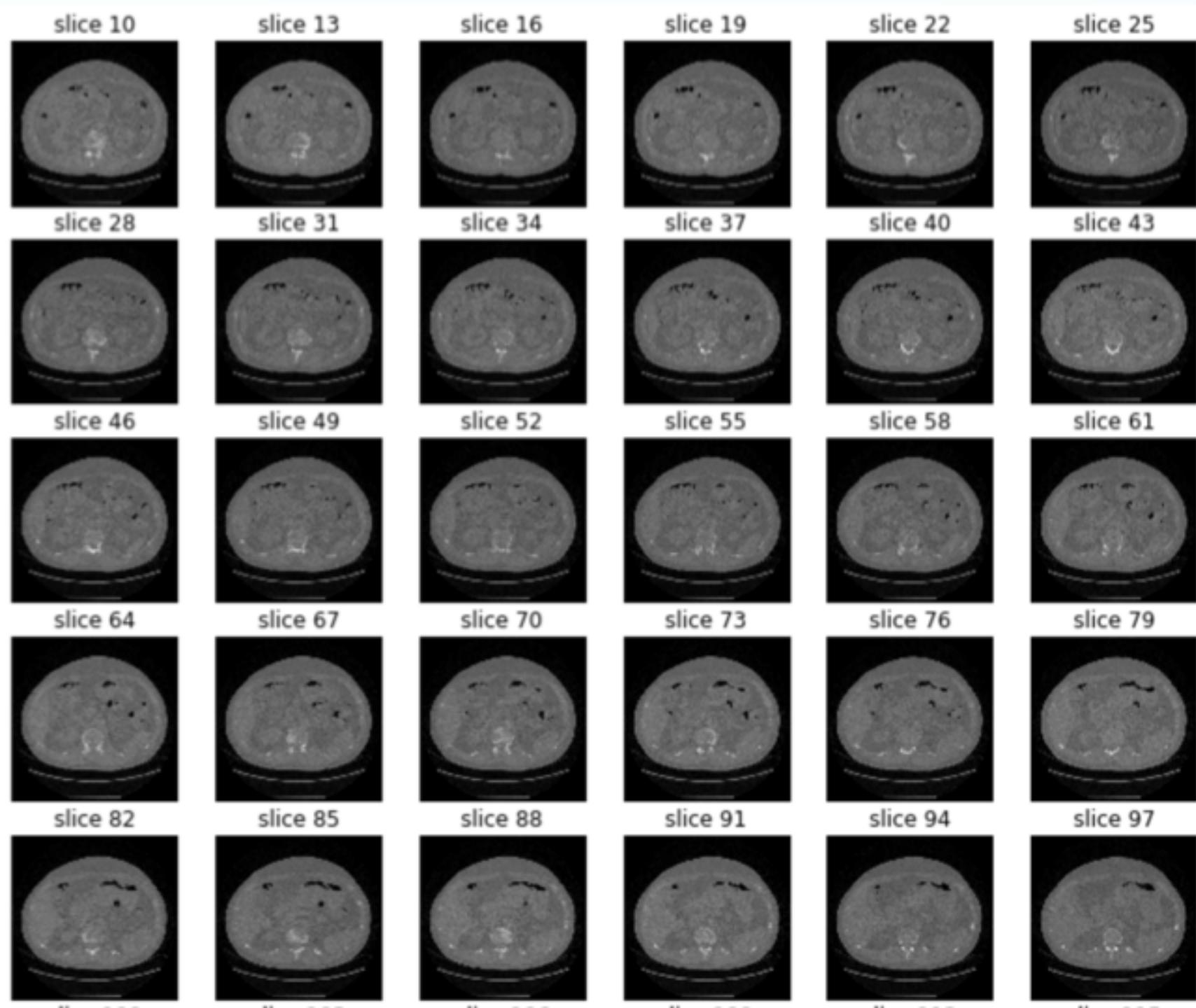


## CT scan visualization and windowing

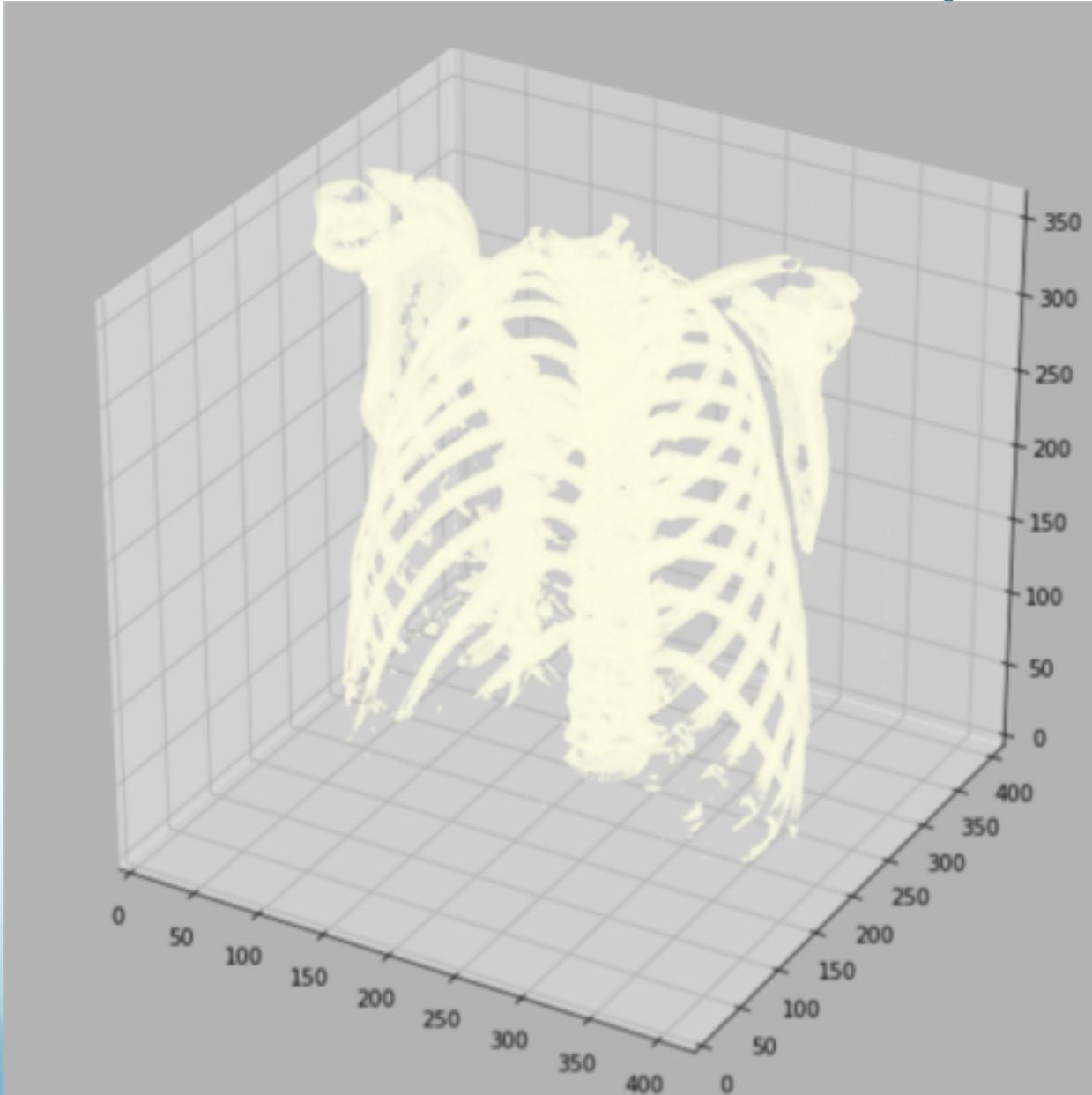
```
In [7]: plt.imshow(dataset.pixel_array, cmap=plt.cm.bone)
plt.show()
```



# CT Slice Visualization



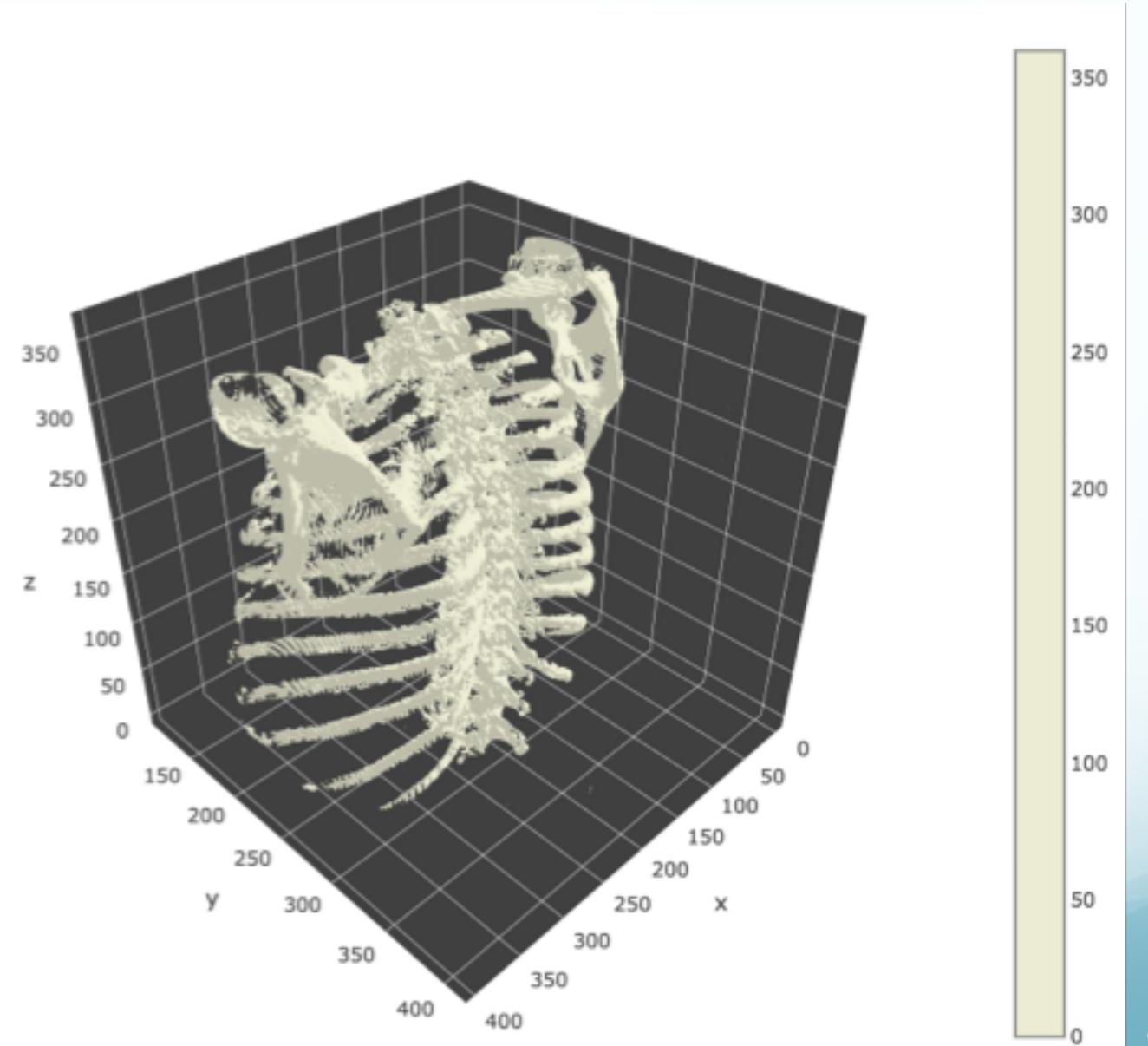
# 3D Reconstruction (1/2)



**Computation time:**  
< 5s on a one-GPU  
AWS instance

# 3D Reconstruction (2/2)

- Interactive view
  - <http://rebrand.ly/copdgene>  
(password protected)



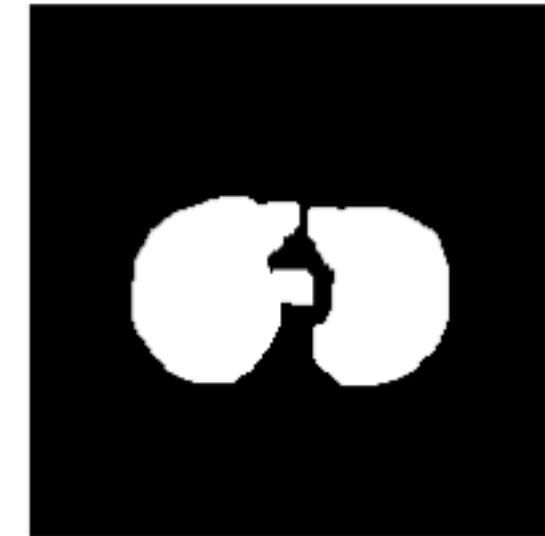
# Cloud-Based Image Segmentation



After Erosion and Dilation



Final Mask



Threshold



Color Labels

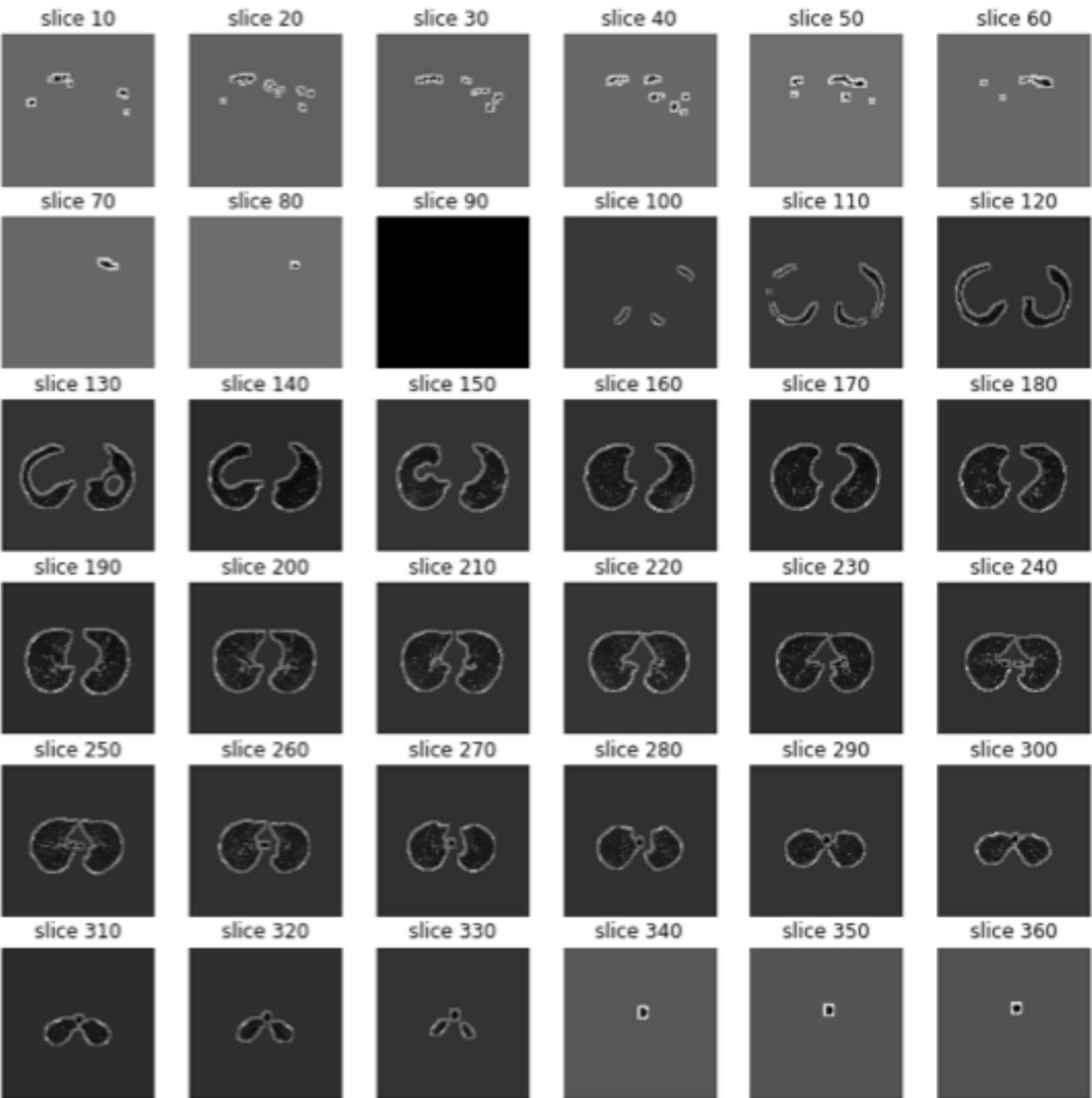


Apply Mask on Original



# Lung Segmentation

- Apply “lung masks” to all slices
  - Computation time:  
< 3s on a one-GPU AWS instance



# Convolutional Neural Networks

- Supports Keras, Tensorflow, PyTorch, and Caffe

```
In [1]: from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras.optimizers import SGD

model = Sequential()

model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3)))
model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(256, (3, 3), activation='relu'))
model.add(Conv2D(256, (3, 3), activation='relu'))
model.add(Conv2D(256, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

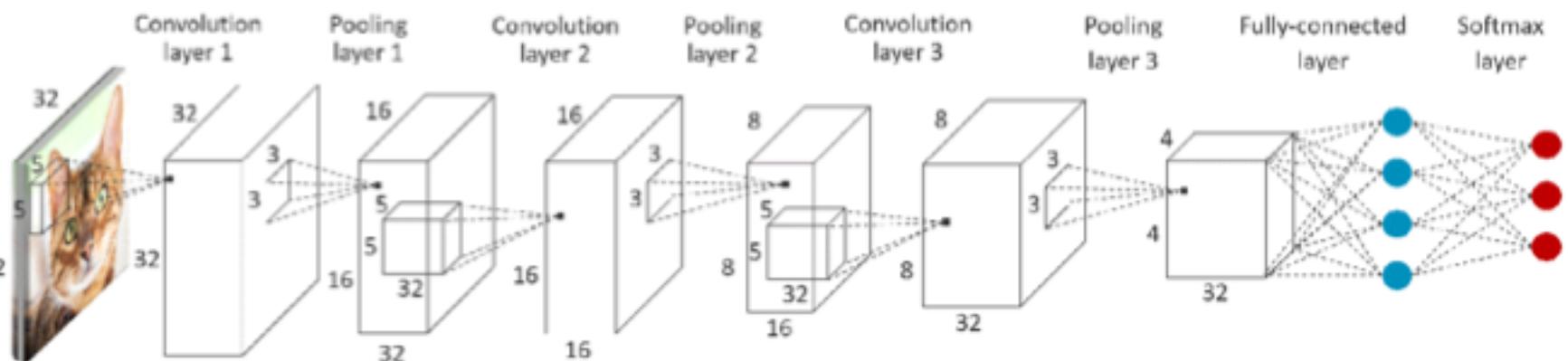
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))
```



Keras

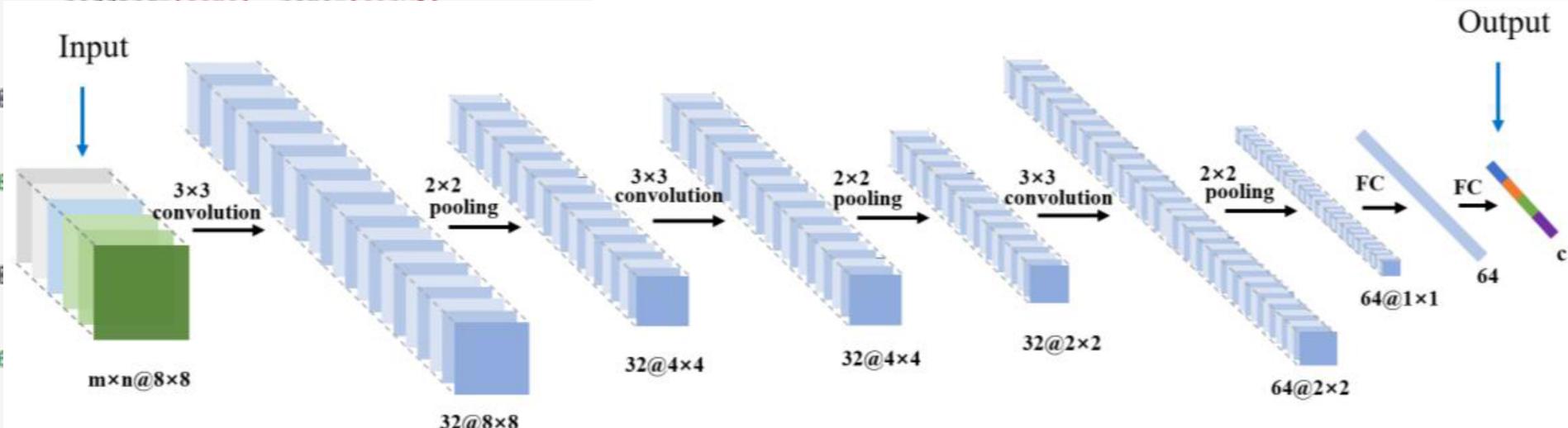
PyTorch

Caffe



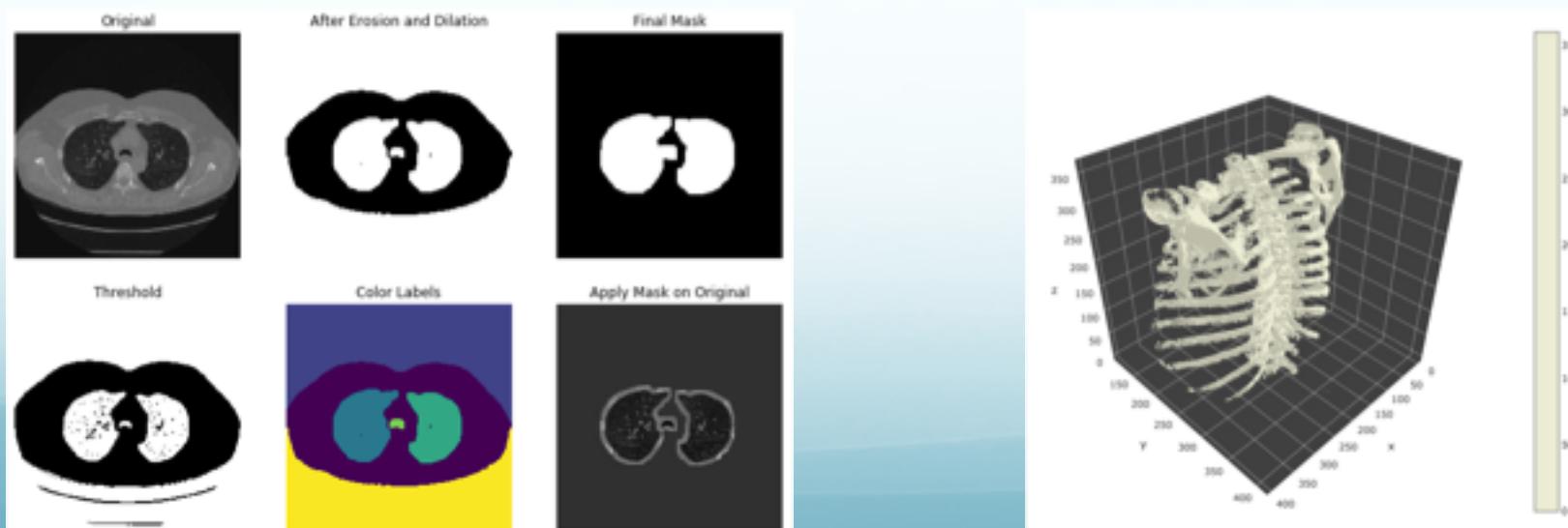
# 3D Convolutional Neural Networks

```
In [1]: model = Sequential()
model.add(Conv3D(64, (3, 3, 3), activation='relu',
                padding='same', name='conv1a',
                subsample=(1, 1, 1)))
model.add(MaxPooling3D(pool_size=(2, 2, 2), strides=(2, 2, 2),
                      padding='valid', name='pool1'))
# 1st layer group
model.add(Conv3D(64, (3, 3, 3), activation='relu',
                padding='same', name='conv1b',
                subsample=(1, 1, 1)))
model.add(MaxPooling3D(pool_size=(2, 2, 2), strides=(2, 2, 2),
                      padding='valid', name='pool2'))
# 2nd layer group
model.add(Conv3D(128, (3, 3, 3), activation='relu',
                 padding='same', name='conv2a',
                 subsample=(1, 1, 1)))
model.add(MaxPooling3D(pool_size=(2, 2, 2), strides=(2, 2, 2),
                      padding='valid', name='pool3'))
# 3rd layer group
model.add(Conv3D(256, (3, 3, 3), activation='relu',
                  padding='same', name='conv3a',
                  subsample=(1, 1, 1)))
model.add(Conv3D(256, (3, 3, 3), activation='relu',
                  padding='same', name='conv3b',
                  subsample=(1, 1, 1)))
model.add(MaxPooling3D(pool_size=(2, 2, 2), strides=(2, 2, 2),
                      padding='valid', name='pool4'))
# 4th layer group
model.add(Conv3D(512, (3, 3, 3), activation='relu',
                  padding='same', name='conv4a',
                  subsample=(1, 1, 1)))
model.add(Conv3D(512, (3, 3, 3), activation='relu',
                  padding='same', name='conv4b',
                  subsample=(1, 1, 1)))
model.add(MaxPooling3D(pool_size=(2, 2, 2), strides=(2, 2, 2),
                      padding='valid', name='pool5'))
# 5th layer group
model.add(Conv3D(512, (3, 3, 3), activation='relu',
                  padding='same', name='conv5a',
                  subsample=(1, 1, 1)))
model.add(Flatten())
model.add(Dense(64))
model.add(Dense(12))
```



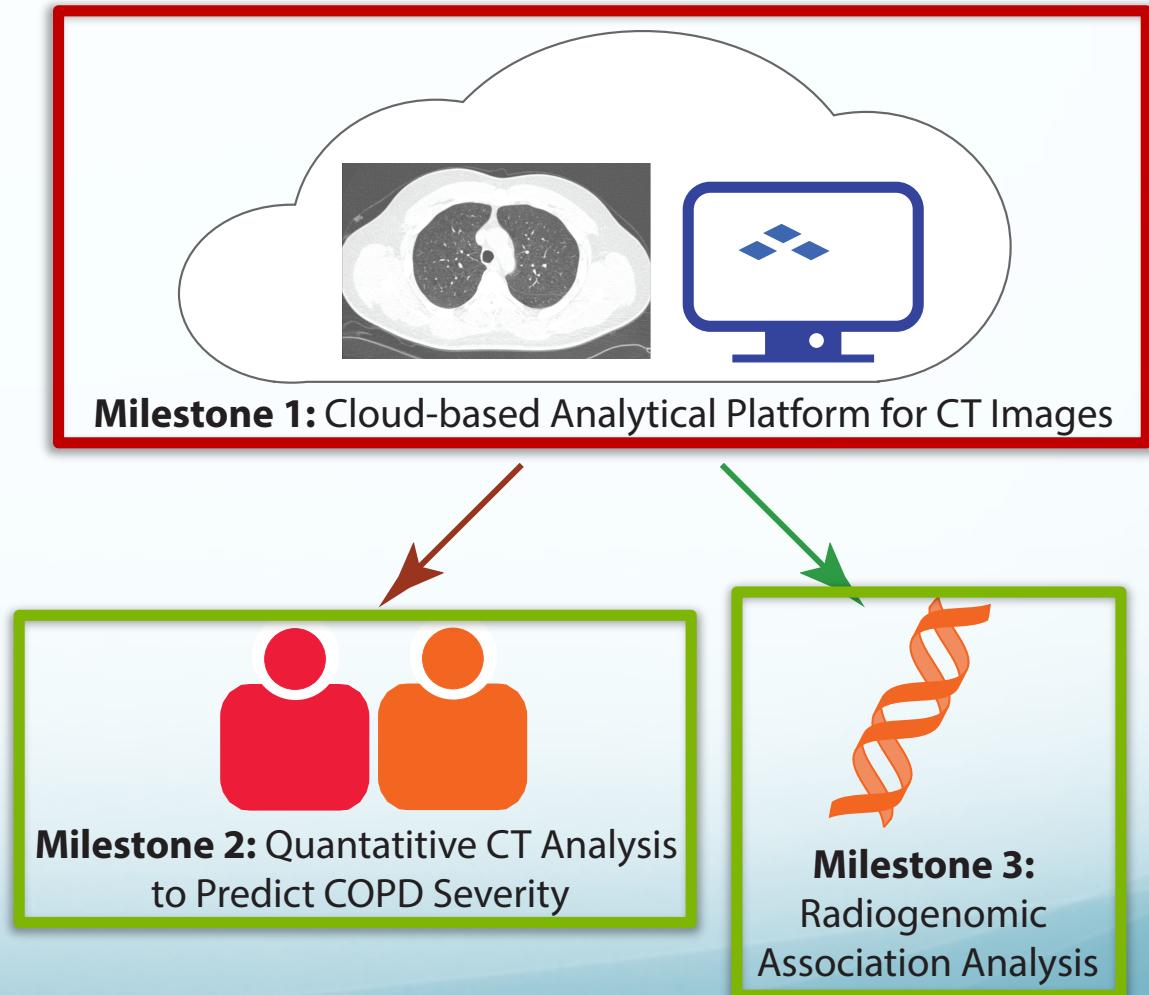
# Summary

- Built a **cloud-based platform** for COPDGene CT images
  - Scalable computing through AWS
    - Data visualization, 3D reconstruction, lung segmentation
    - 2D and 3D convolutional neural networks
    - Deployed an Amazon Machine Image (AMI): **ami-7d6e4502**
  - Platform validated on Google Cloud and Microsoft Azure Cloud



# Future Works

- To build convolutional neural network models using this framework
  - To predict COPD severity via quantitative CT analysis
  - To identify novel radio-genomic associations using the COPDgene cohort



# Acknowledgements

- Zak Lab
  - Zak Kohane, MD, PhD
  - Arjun Manrai, PhD
- Avillach Lab
  - Paul Avillach, MD, PhD
  - Gregoire Versmee, MD
  - Laura Versmee, MD
- Harvard DBMI
  - Susanne Churchill, PhD
  - Chirag Patel, PhD
- Funding



# Thank you. 😊

Kun-Hsing\_Yu@hms.harvard.edu

