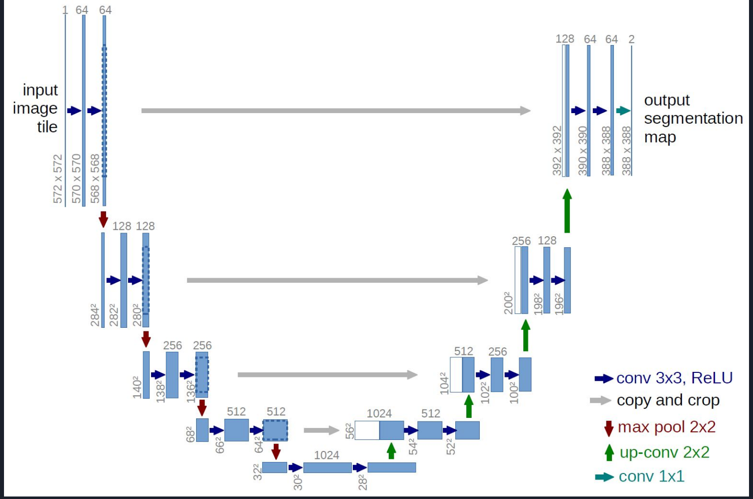
Semantic Segmentation

Team C:
Ryan Good, Aidan Carter Scott, Anthony Burch, Neil Thupili

Client: Andrew Abumoussa, MD

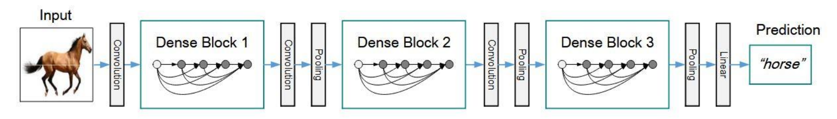
Platform

- Our users: Surgeons, medical professionals
- Our goal: Use machine learning as a tool to process radiographic images, classify their contents into useful features
- Built in Python, using TensorFlow
- Flask front-end, but client just wants the model
- Transfer learning approach



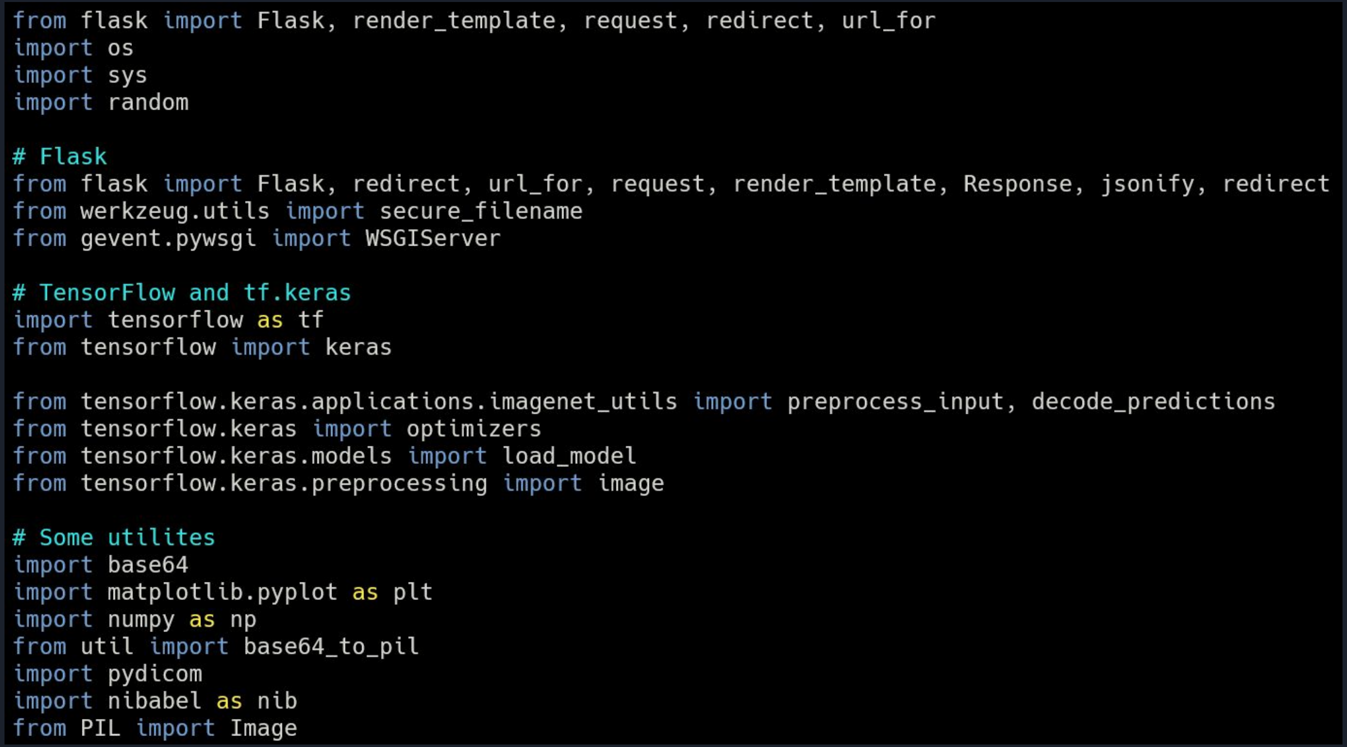
Ronneberger, O., Fischer, P., Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science(), vol 9351. Springer, Cham.

- U-Net: Convolutional Neural Network (CNN) for biomedical image segmentation
- DenseNet-121: Densely connected CNN, every layer is connected to every deeper layer. 121 layers total
- Benefit of transfer learning: we do not need to know how these work to use them



G. Huang, Z. Liu, L. Van Der Maaten and K. Weinberger, "Densely Connected Convolutional Networks," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017 pp. 2261-2269.

What we used

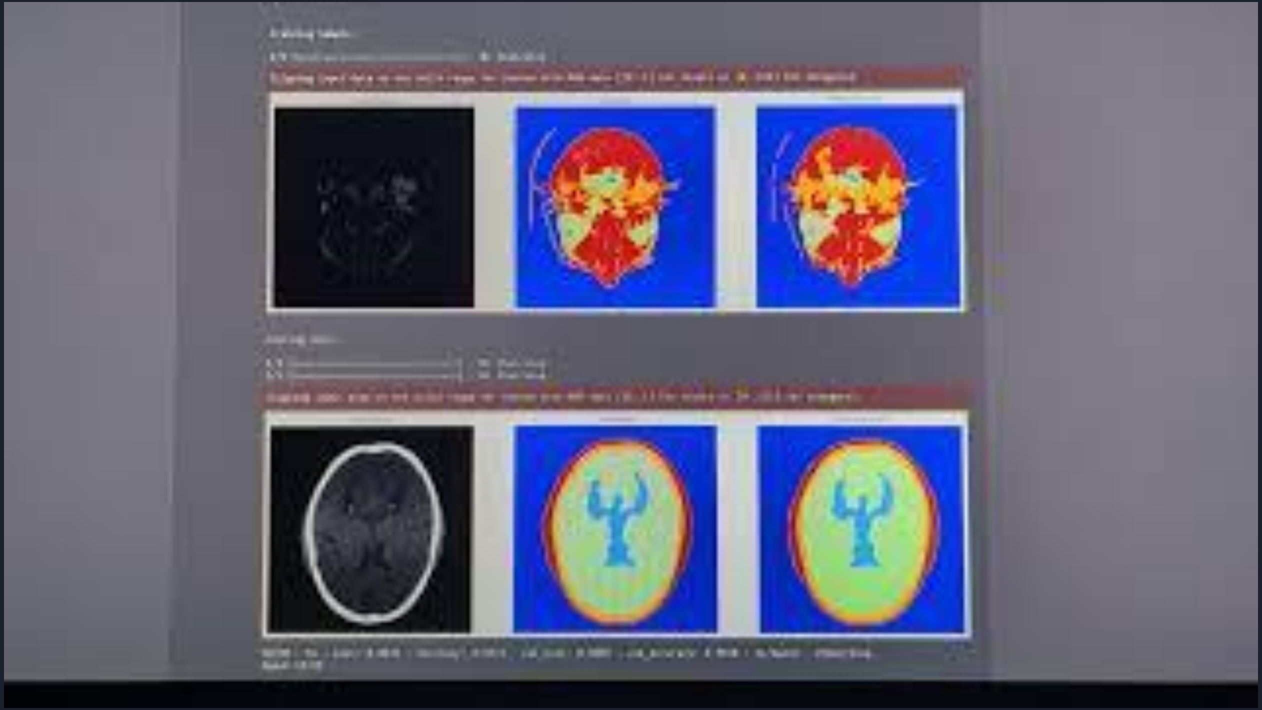


```
from flask import Flask, render_template, request, redirect, url_for
import os
import sys
import random

# Flask
from flask import Flask, redirect, url_for, request, render_template, Response, jsonify, redirect
from werkzeug.utils import secure_filename
from gevent.pywsgi import WSGIServer

# TensorFlow and tf.keras
import tensorflow as tf
from tensorflow import keras

from tensorflow.keras.applications.imagenet_utils import preprocess_input, decode_predictions
from tensorflow.keras import optimizers
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image

# Some utilites
import base64
import matplotlib.pyplot as plt
import numpy as np
from util import base64_to_pil
import pydicom
import nibabel as nib
from PIL import Image
```

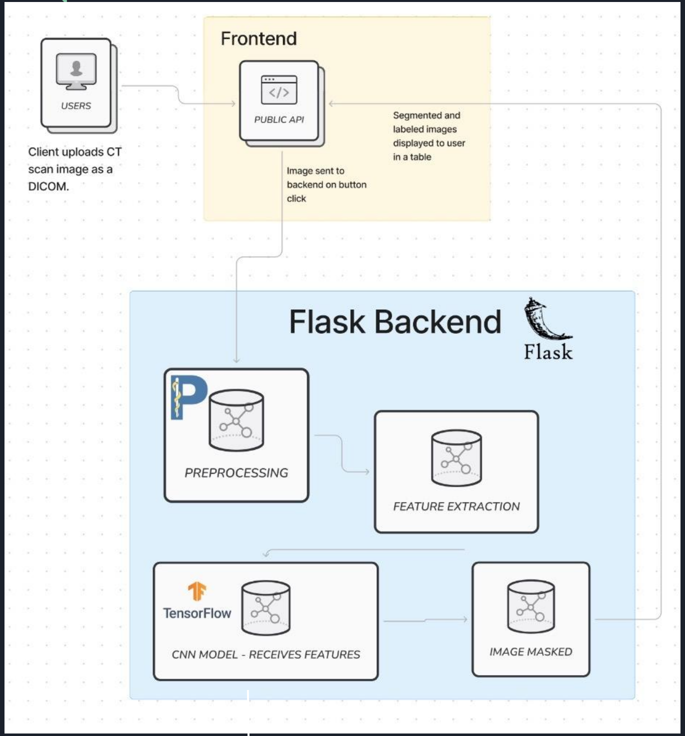
Demo

- One moment to set up…

Demo



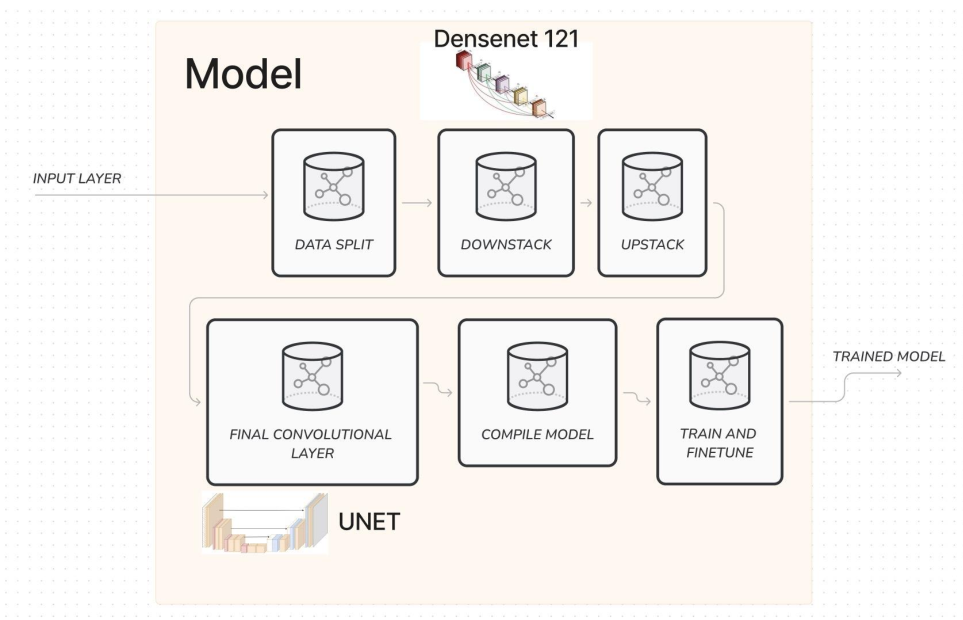
Architecture



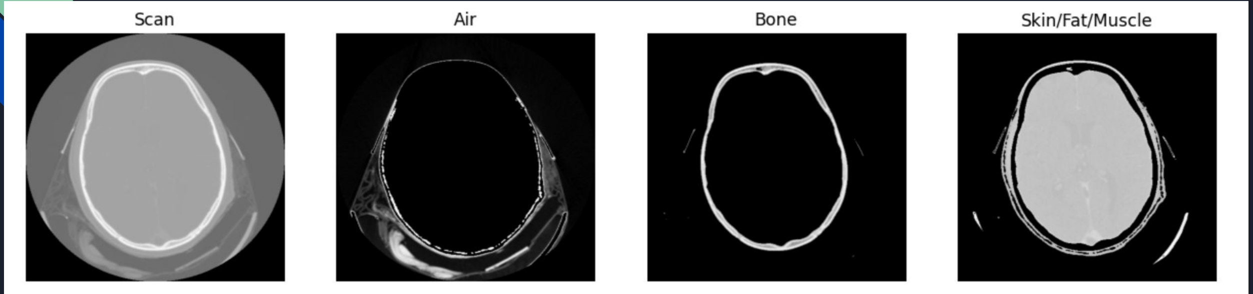
Client uploads CT scan image as a DICOM.

Image sent to backend on button click

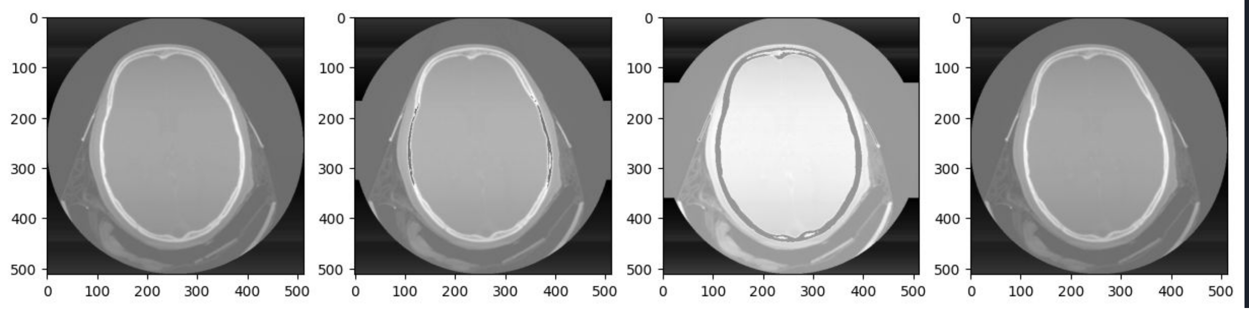
Segmented and labeled images displayed to user in a table



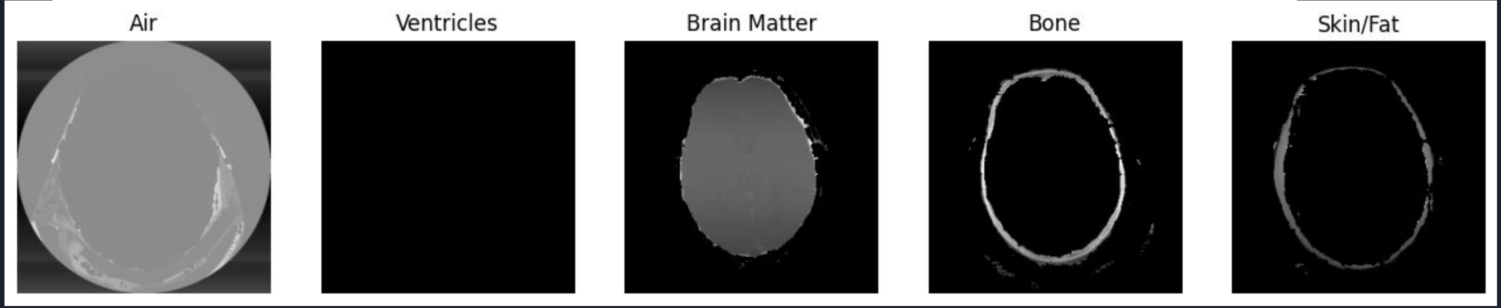
Lessons Learned



Simple Masking (unclustered)



K-Means (clustered)



Model Results

Lessons Learned

- Any data manipulation has to be done to all the training data as well
  - PIL image -> Numpy array -> Tensor , major issues
- Training data should be diverse
  - Also include confusing data
- Transfer learning is immensely helpful
- The field is niche, in weird ways