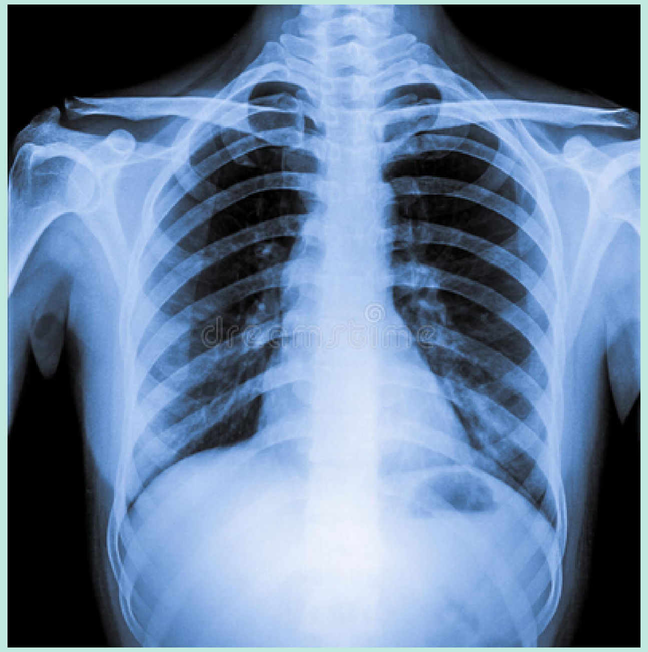
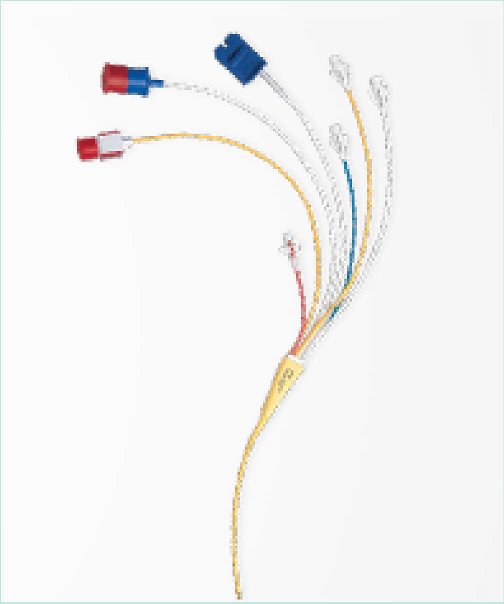
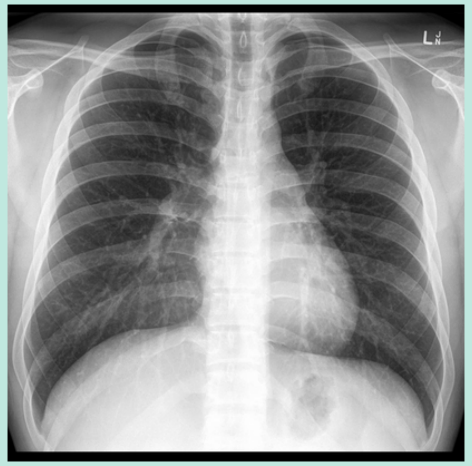
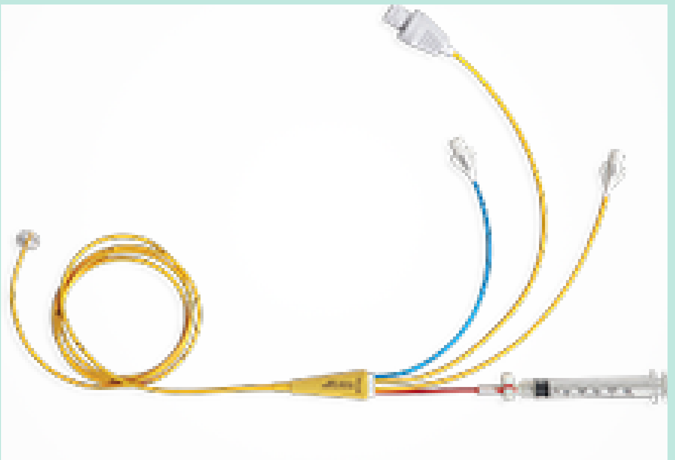








AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

CATHETER AND LINE POSITIONING IN CHEST X-RAYS

CALP-X

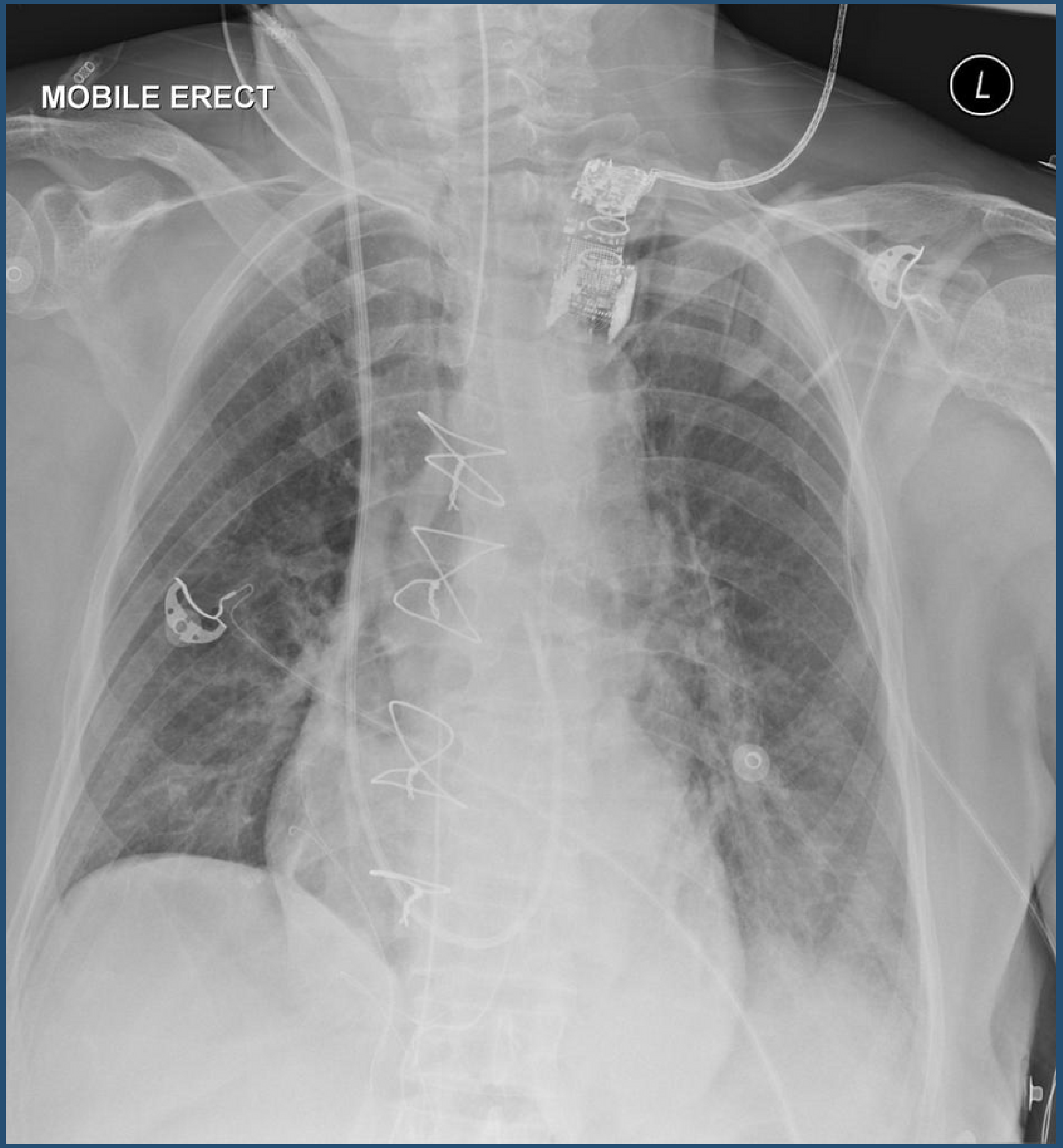
Final Presentation, Team 7

Team Details

K V S CHATHURYA BL.EN.U4CSE18050
M L S CHANDRA MOULIKA BL.EN.U4CSE18062
MAKKENA VAISHNAVI BL.EN.U4CSE18063
MANASA SANJEEV BL.EN.U4CSE18065

Agenda

- Introduction / Abstract
- Problem statement
- Study on the Existing System
- Design methodology
- Modules each individual
- Implementation & Demo
- Conclusion / Future Scope
- Timeline chart
- Learning
- References



INTRODUCTION & ABSTRACT

- Catheters are life-saving equipment.
- Mal positioning implies serious complications and can even be fatal.
- Early recognition is the key to solve this problem.
- Use medical imaging to train a deep learning model.
- Classify the Catheters into 4 main sub-classes.

Problem Statement

- Detect the presence and position of catheters and lines on chest x-rays.
- Use deep learning pretrained model to test on 3000 images to categorize different positions of catheters.
- To alert the radiologists in case of mispositioning.

EXISTING SYSTEM

- At present, no automated system exists for the same.
- Chest radiograph (CXR) plays a crucial role in evaluating the position.
- Other imaging mechanisms like CT scans and MRIs haven't been perfected to be used for the same cause.
- Radiologists are trained to accurately perform the task with minimum error.
- Time lag (latency) between the production of images and the true position of the catheter.
- Catheters may be confused by other similar linear structures like ECG leads and anatomy including ribs.

Model Design

Research

Importing Libraries

EDA

Image Transformations

Prepare DataLoader

Importing Pretrained Model

Inference of Test Dataset

Result

Individual Contribution

Chathurya

Research on type of pre-trained CNNs, EDA, SeResnet152D

Manasa

Research on catheters and existing system, EDA, Resnet50

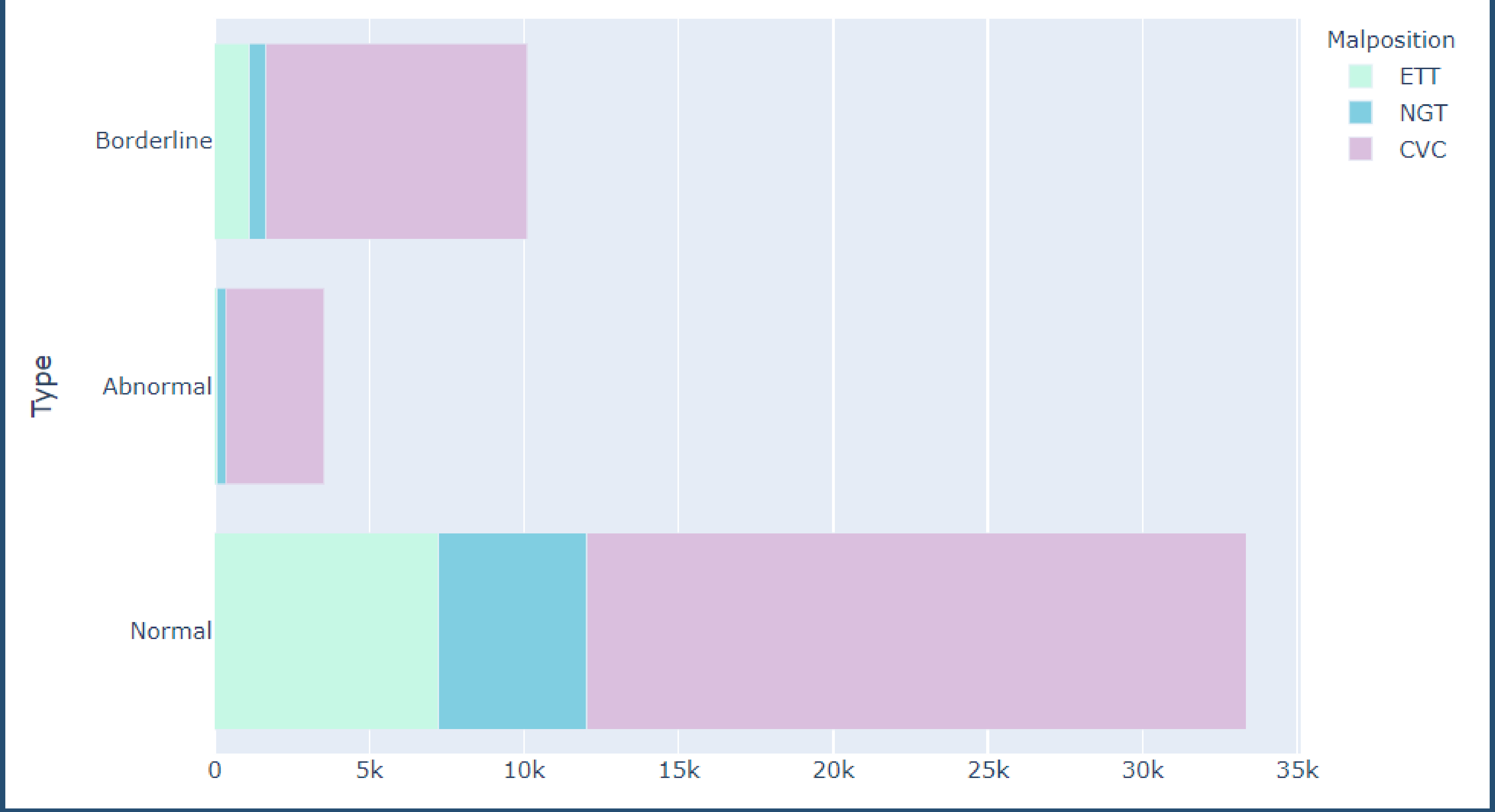
Vaishnavi

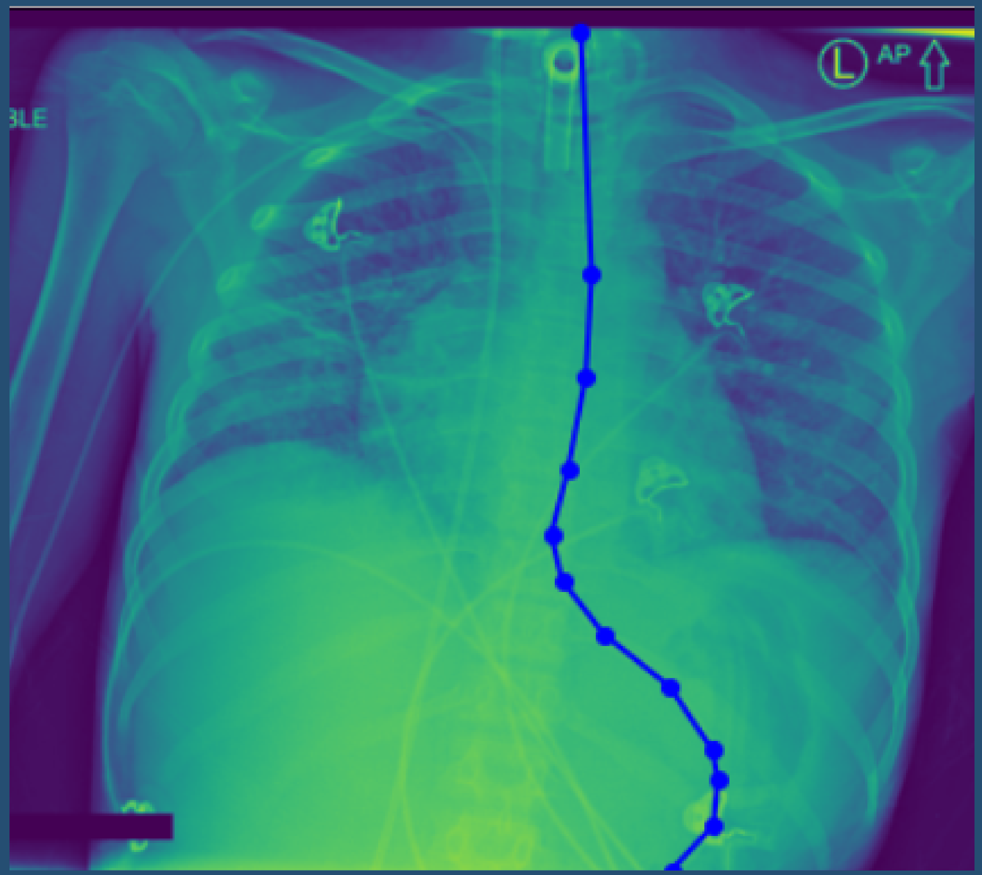
Research on catheters, EDA, Resnet 200D

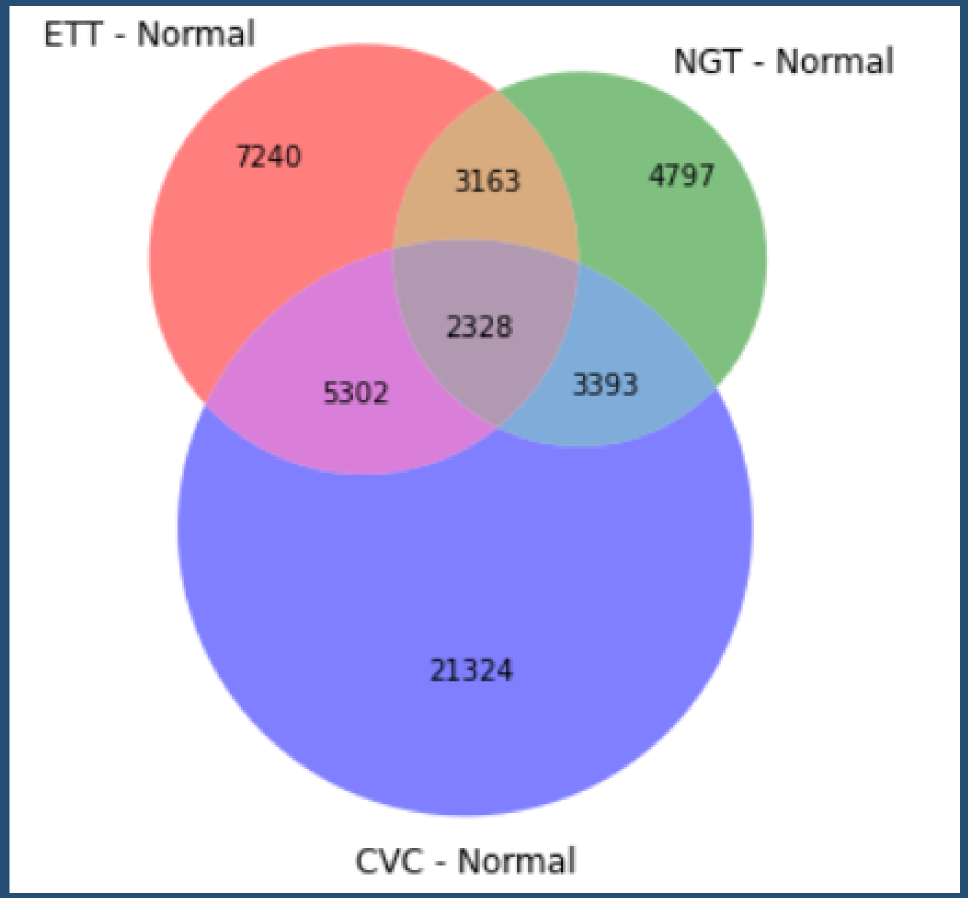
Moulika

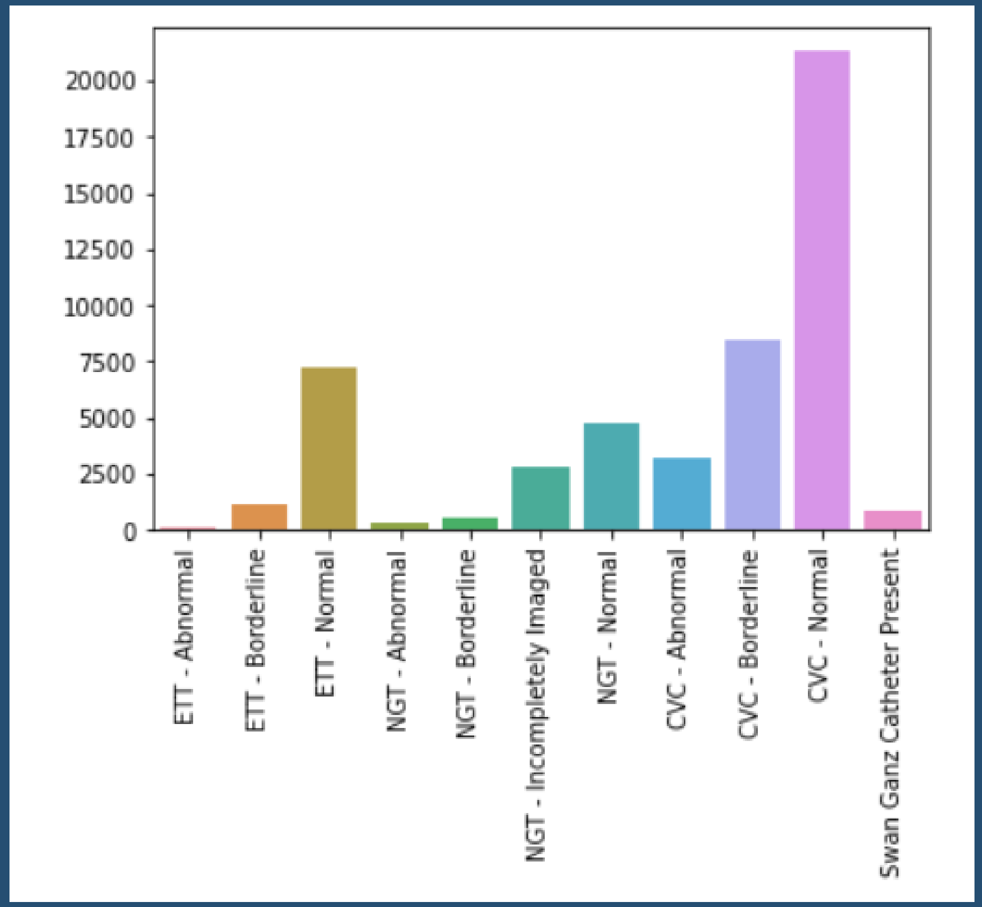
Research on peer review and Streamlit app, EDA, Resnet50

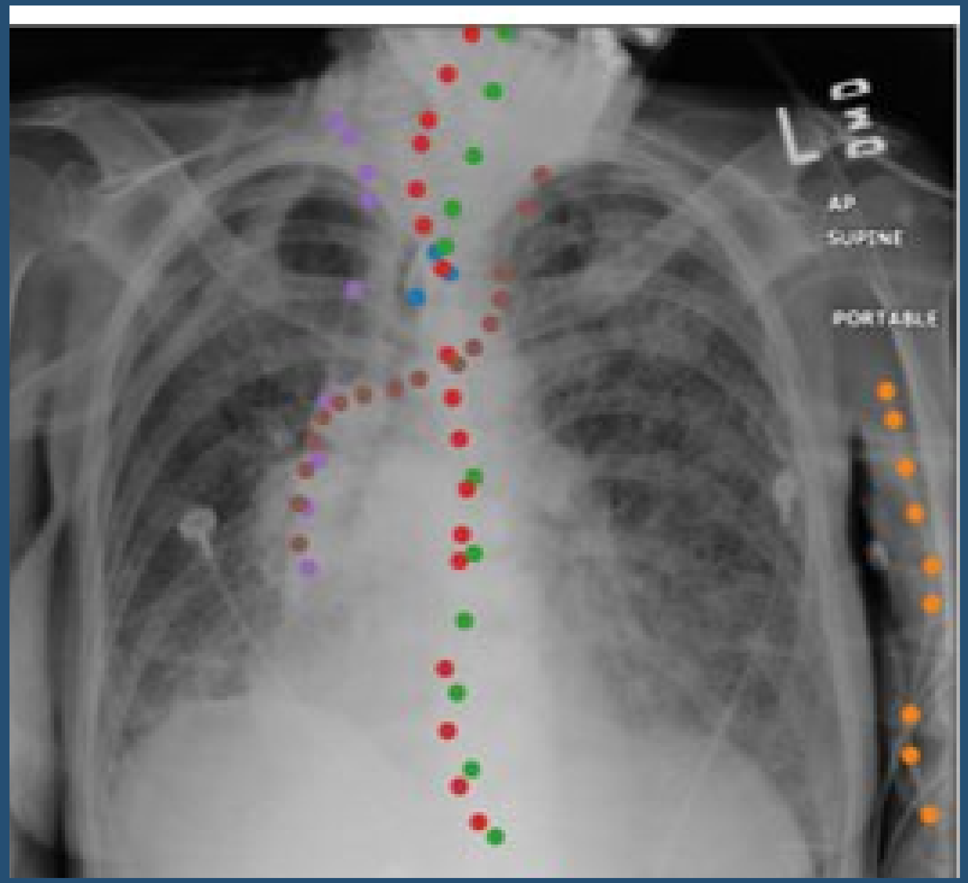
Exploratory Data Analysis




L
AP

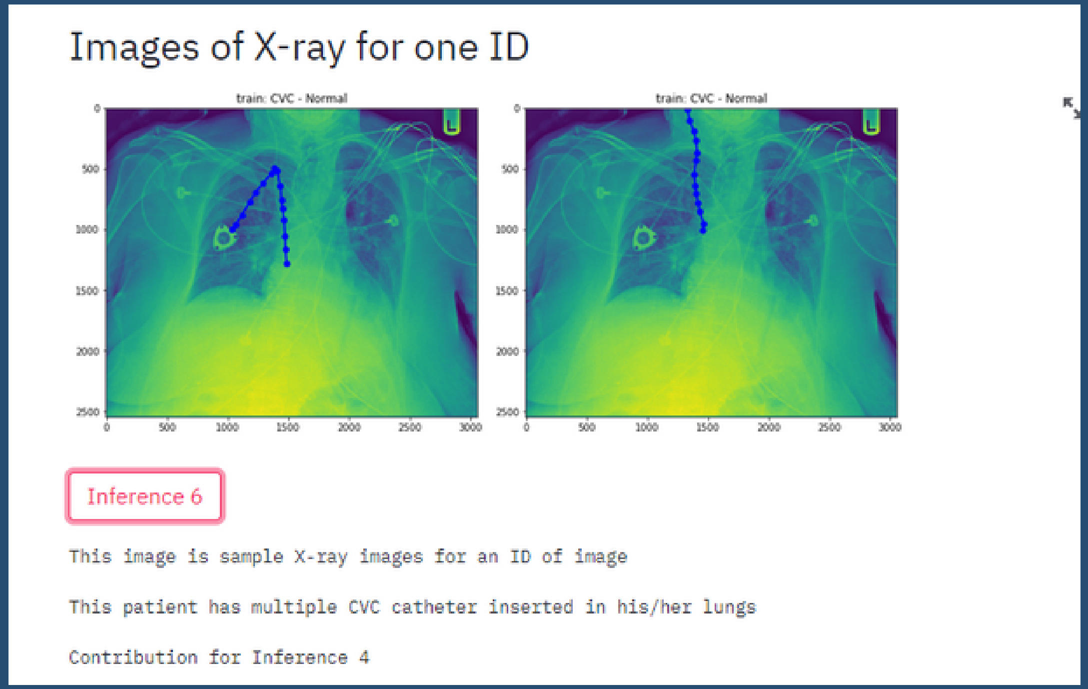

ETT - Normal
NGT - Normal
CVC - Normal
7240
3163
4797
2328
5302
3393
21324


20000
17500
15000
12500
10000
7500
5000
2500
0
ETT - Abnormal
ETT - Borderline
ETT - Normal
NGT - Abnormal
NGT - Borderline
NGT - Incompletely Imaged
NGT - Normal
CVC - Abnormal
CVC - Borderline
CVC - Normal
Swan Ganz Catheter Present

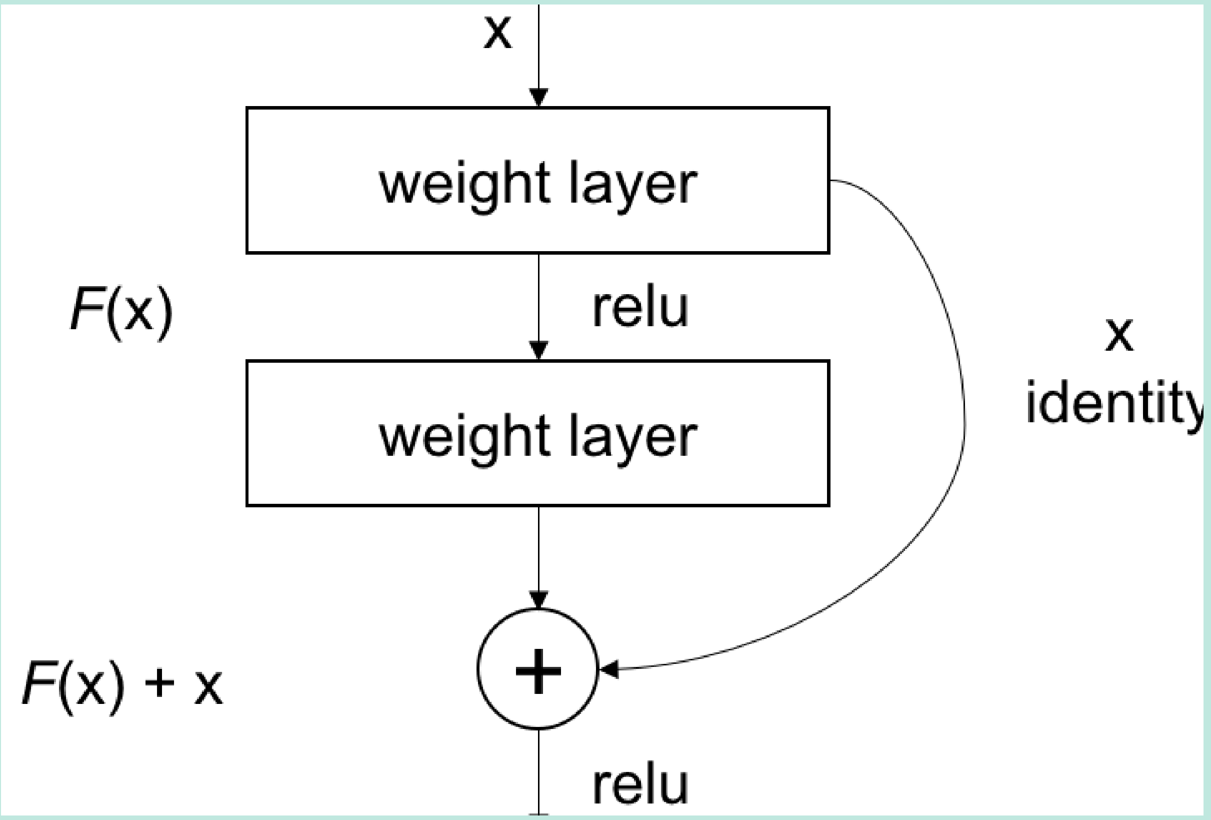

AP
SUPINE
PORTABLE

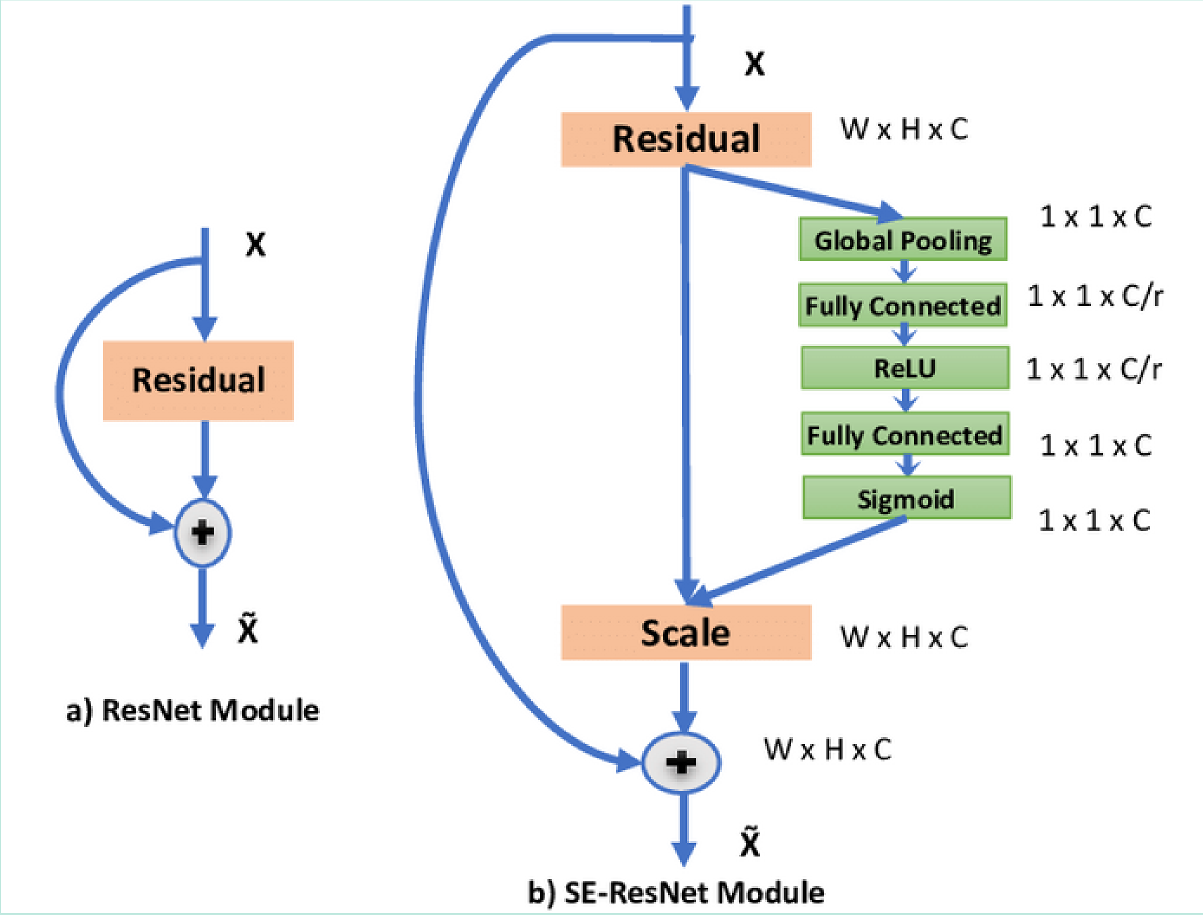
Streamlit

Built an streamlit app



EDA Analysis on Streamlit app


x
weight layer
F(x)
relu
x
identity
weight layer
F(x) + x
relu


x
Residual
x̃
a) ResNet Module
x
Residual
W x H x C
Global Pooling
1 x 1 x C
Fully Connected
1 x 1 x C/r
ReLU
1 x 1 x C/r
Fully Connected
1 x 1 x C
Sigmoid
1 x 1 x C
Scale
W x H x C
W x H x C
x̃
b) SE-ResNet Module

MODEL

Functions

PREPARE TEST DATASET

```
class TestDataset(Dataset):
    def __init__(self, df, transform=None):
        self.df = df
        self.file_names = df['StudyInstanceUID'].values
        self.transform = transform
```

IMAGE TRANSFORMATIONS

```
def get_transforms():
        return Compose([
            Resize(IMAGE_SIZE, IMAGE_SIZE),
            Normalize(
            ),
            ToTensorV2(),
        ])
```

MODEL

```
class CustomResNext(nn.Module):
    def __init__(self, model_name='resnext50_32x4d', pretrained=False):
        super().__init__()
        self.model = timm.create_model(model_name, pretrained=pretrained)
        n_features = self.model.fc.in_features
        self.model.fc = nn.Linear(n_features, CFG.target_size)

    def forward(self, x):
        x = self.model(x)
        return x
```

INFERENCE

```
inference(models, test_loader, device):
tk0 = tqdm(enumerate(test_loader), total=len(test_loader)) ##progress Bar
probs = []
for i, (images) in tk0:
    images = images.to(device)
    avg_preds = []
    for model in models:
        with torch.no_grad():
            y_preds1 = model(images)
            y_preds2 = model(images.flip(-1))
        y_preds = (y_preds1.sigmoid().to('cpu').numpy() + y_preds2.sigmoid().to('cpu').numpy()) / 2
        avg_preds.append(y_preds)
    avg_preds = np.mean(avg_preds, axis=0)
    probs.append(avg_preds)
probs = np.concatenate(probs)
return probs
```

Resnet50

```python
class CustomResNext(nn.Module):
    def __init__(self, model_name='resnext50_32x4d', pretrained=False):
        super().__init__()
        self.model = timm.create_model(model_name, pretrained=pretrained)
        n_features = self.model.fc.in_features
        self.model.fc = nn.Linear(n_features, CFG.target_size)

    def forward(self, x):
        x = self.model(x)
        return x
```

StudyInstanceUID	ETT - Abnormal	ETT - Borderline	ETT - Normal	NGT - Abnormal	NGT - Borderline	NGT - Incompletely Imaged	NGT - Normal	CVC - Abnormal	CVC - Borderline	CVC - Normal	Swan Ganz Catheter Present
043.8.498.46923145579096002617...	0.059688	0.577791	0.216742	0.000811	0.010068	0.006182	0.985213	0.031307	0.148838	0.955181	0.998979
043.8.498.84006870182611080091...	0.000051	0.000206	0.000524	0.000288	0.000850	0.000441	0.000092	0.001722	0.008559	0.998135	0.000044
043.8.498.12219033294413119947...	0.000039	0.000121	0.000502	0.000869	0.000471	0.000259	0.000064	0.009284	0.569036	0.653767	0.000054
043.8.498.84994474380235968109...	0.005164	0.026795	0.058099	0.045763	0.026076	0.952735	0.035707	0.158107	0.164486	0.927931	0.011512
043.8.498.35798987793805669662...	0.000292	0.000997	0.001352	0.001461	0.005168	0.000082	0.001627	0.003232	0.045860	0.973941	0.000076

SeResnet152D

```python
class SeResNet152D(nn.Module):
    def __init__(self, model_name='seresnet152d_320'):
        super().__init__()
        self.model = timm.create_model(model_name, pretrained=False)
        n_features = self.model.fc.in_features
        self.model.global_pool = nn.Identity()
        self.model.fc = nn.Identity()
        self.pooling = nn.AdaptiveAvgPool2d(1)
        self.fc = nn.Linear(n_features, 11)

    def forward(self, x):
        bs = x.size(0)
        features = self.model(x)
        pooled_features = self.pooling(features).view(bs, -1)
        output = self.fc(pooled_features)
        return output
```

```python
ser_model = SeResNet152D()
ser_model.load_state_dict(torch.load(SER_MODEL_PATH)['model'])
```

Out[18]:

StudyInstanceUID	ETT - Abnormal	ETT - Borderline	ETT - Normal	NGT - Abnormal	NGT - Borderline	NGT - Incompletely Imaged	NGT - Normal	CVC - Abnormal	CVC - Borderline	CVC - Normal	Swan Ganz Catheter Present
43.8.498.46923145579096002617...	0.002861	0.186532	0.791820	0.000648	0.002636	0.017136	0.980327	0.023258	0.084417	0.947521	9.992568e-01
43.8.498.84006870182611080091...	0.000002	0.000025	0.000052	0.000016	0.000016	0.000006	0.000010	0.021782	0.004071	0.984555	5.294564e-07
43.8.498.12219033294413119947...	0.000023	0.000074	0.000149	0.000226	0.000193	0.000202	0.000278	0.014617	0.471231	0.485876	3.295735e-05
43.8.498.84994474380235968109...	0.005439	0.067142	0.094417	0.077373	0.019847	0.876966	0.039560	0.060899	0.078361	0.737870	2.001944e-03
43.8.498.35798987793805669662...	0.000378	0.001077	0.000824	0.005594	0.001927	0.000676	0.002276	0.007787	0.074417	0.907452	3.312462e-06

Resnet200D

```python
class ResNet200D(nn.Module):
    def __init__(self, model_name='resnet200d_320'):
        super().__init__()
        self.model = timm.create_model(model_name, pretrained=False)
        n_features = self.model.fc.in_features
        self.model.global_pool = nn.Identity()
        self.model.fc = nn.Identity()
        self.pooling = nn.AdaptiveAvgPool2d(1)
        self.fc = nn.Linear(n_features, 11)

    def forward(self, x):
        bs = x.size(0)
        features = self.model(x)
        pooled_features = self.pooling(features).view(bs, -1)
        output = self.fc(pooled_features)
        return output
```

```python
res_model = ResNet200D()
res_model.load_state_dict(torch.load(RES_MODEL_PATH)['model'])
```

StudyInstanceUID	ETT - Abnormal	ETT - Borderline	ETT - Normal	NGT - Abnormal	NGT - Borderline	NGT - Incompletely Imaged	NGT - Normal	CVC - Abnormal	CVC - Borderline	CVC - Normal	Swan Ganz Catheter Present
043.8.498.46923145579096002617...	0.042606	0.621123	0.217875	0.000425	0.003781	0.011037	0.980665	0.037586	0.140550	0.889836	0.999464
043.8.498.84006870182611080091...	0.000010	0.000032	0.000242	0.000163	0.000110	0.000041	0.000067	0.004692	0.004040	0.998147	0.000002
043.8.498.12219033294413119947...	0.000016	0.000017	0.000109	0.000139	0.000109	0.000012	0.000033	0.005357	0.241475	0.770350	0.000017
043.8.498.84994474380235968109...	0.001823	0.020463	0.044035	0.022296	0.011816	0.919738	0.031636	0.049662	0.051236	0.913974	0.000491
043.8.498.35798987793805669662...	0.000026	0.000189	0.001200	0.001190	0.000670	0.000095	0.002048	0.003164	0.006033	0.991957	0.000002

Conclusion & Future Scope

A successful model has been built

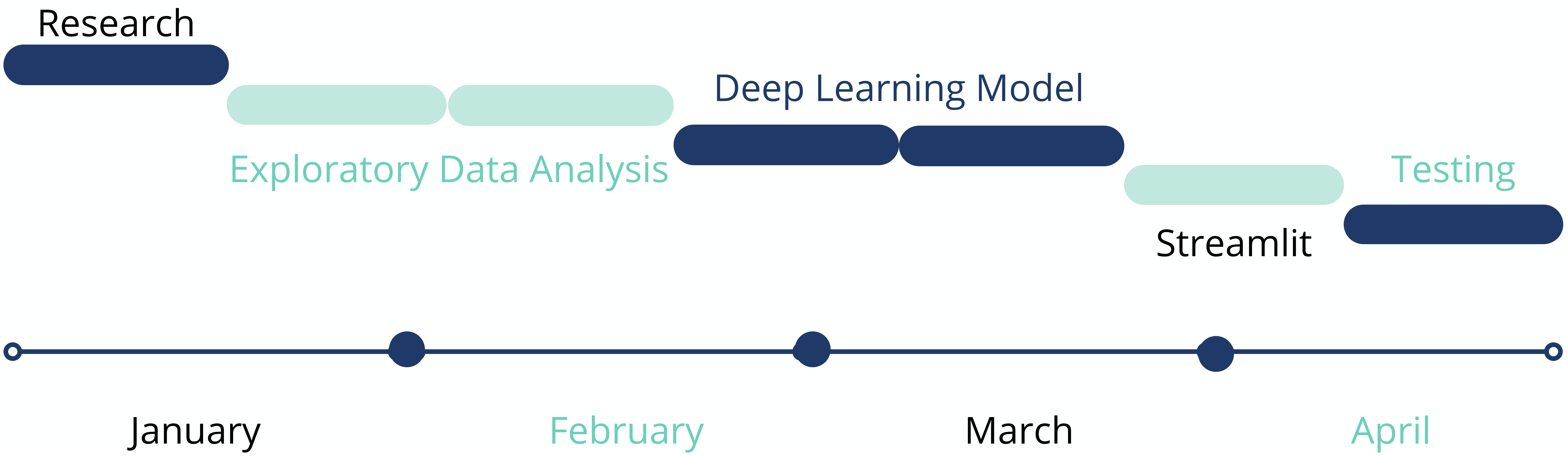
The dataset was run on a pre-trained model

Can use GPU to improve accuracy

GPU- Graphical Processing Unit

Launch an App

We can pickle the trained model and run it on an app using Streamlit.



TIMELINE CHART

LEARNING

- Pandas
- Pytorch
- Streamlit
- Pickle
- Matplotlib
- Seaborn
- Different types of pre-trained models of CNN

REFERENCES

- https://www.kaggle.com/c/ranzcr-clip-catheter-line-classification/overview
- https://docs.google.com/spreadsheets/d/1oWUL60YnAAiGD6-qLYukvk4FmY1qVU3r-58eMterhcE/edit?usp=sharing

- Please note that the google sheets link above contains the rest of the links.

THANKYOU