Online Skin Lesion Classification with Deep Neural Networks

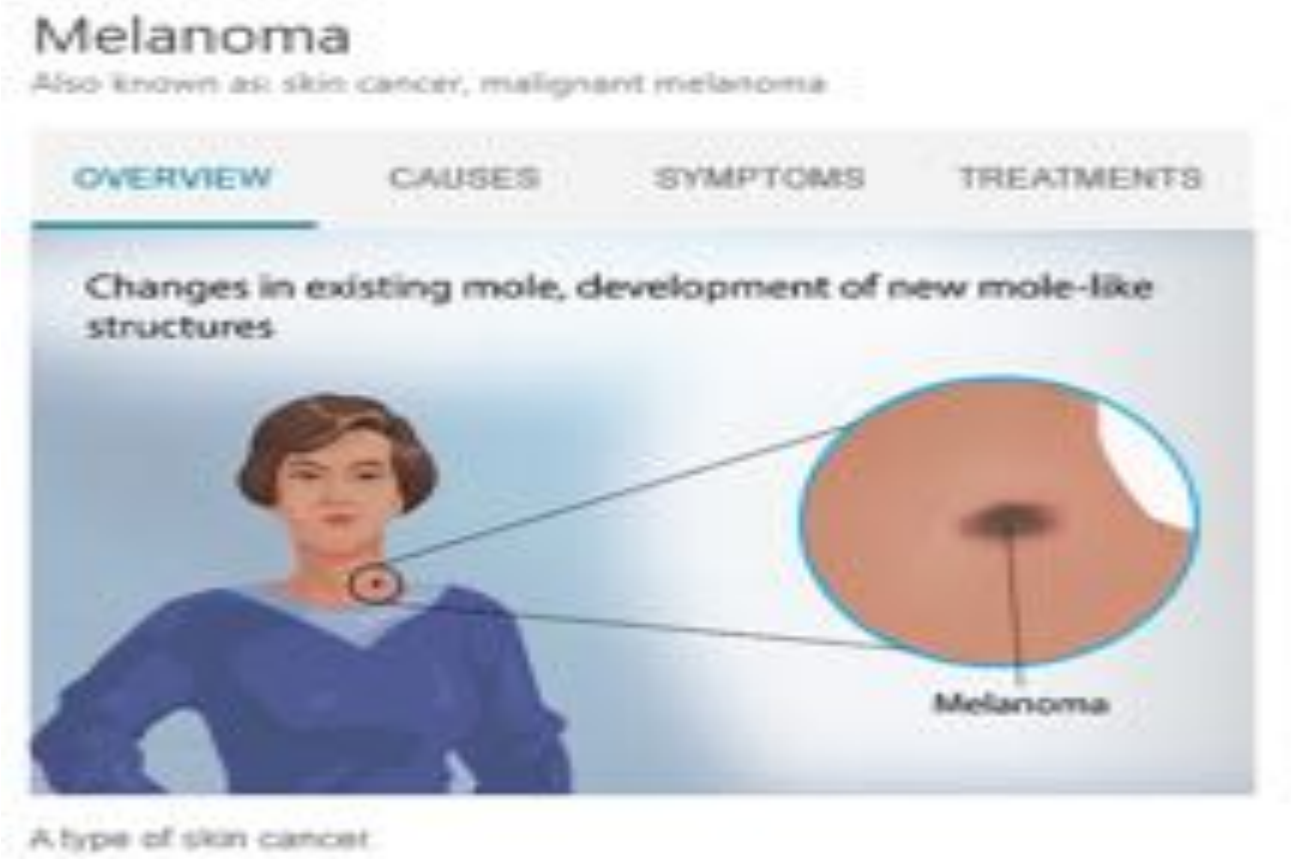
PRESENTED BY VICTORIA PHAM

AUGUST 15TH 2019

UNIVERSITY OF PENNSYLVANIA

MENTOR: HAMSA BASTANI & ARIELLE ANDERER

Melanoma


Melanoma
Also known as: skin cancer, malignant melanoma
OVERVIEW
CAUSES
SYMPTOMS
TREATMENTS
Changes in existing mole, development of new mole-like structures
Melanoma
A type of skin cancer

Machine Learning

What is Machine Learning?

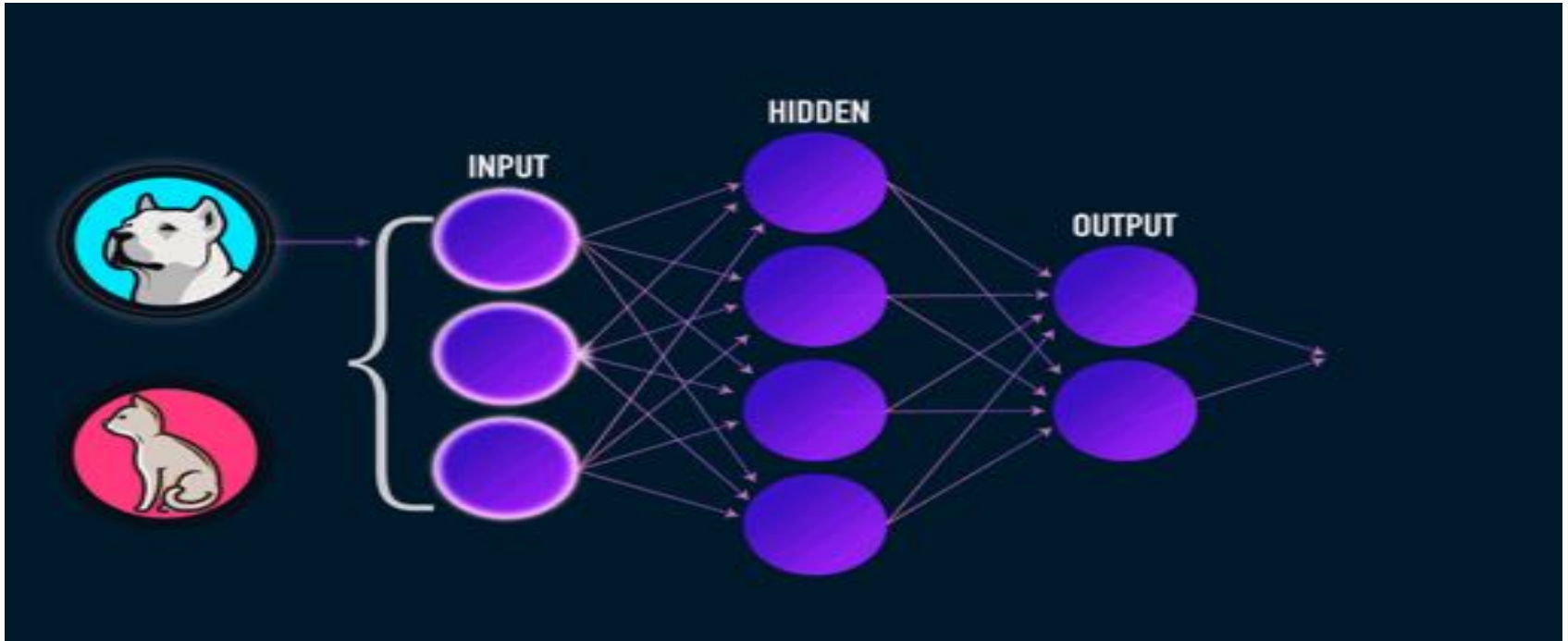
Learn From Experience

Learn From ~~Experience~~ Data

Follow Instructions

Deep Learning

- Deep learning's sophisticated technology and self-learning capabilities result in higher accuracy and faster processing.
- Raw data is fed through deep neural networks, which learns to identify the object on which it is trained

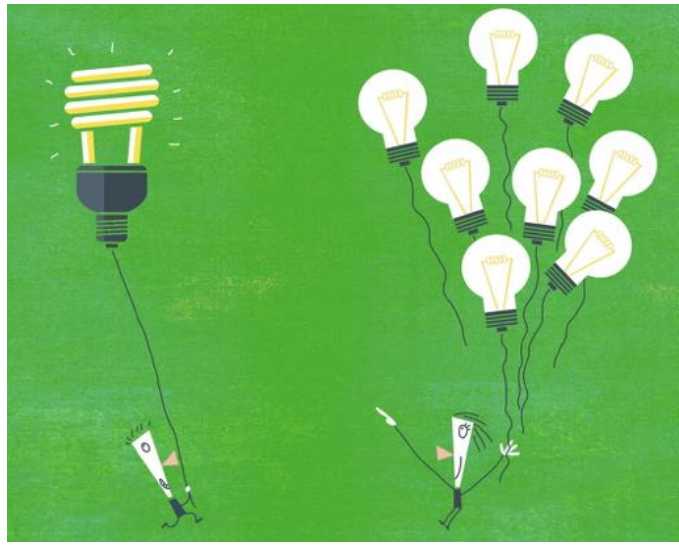


Research Goals

- Developing an algorithm that pre-screens people for cancerous melanoma in places where resources for in-person testing is limited
- There are already algorithms that classify well on images taken in the U.S. at a standard distance and focus.

Significance

- **Saving Time & Cost**: Providing more effectively resources to those who truly need to travel long distance to meet in person with doctors
- Clinics and doctors can **allocate resources wisely**
- Making screening **less time-intensive**
- **Accessible** to those who are truly in need (especially in resource limited country)



Methods

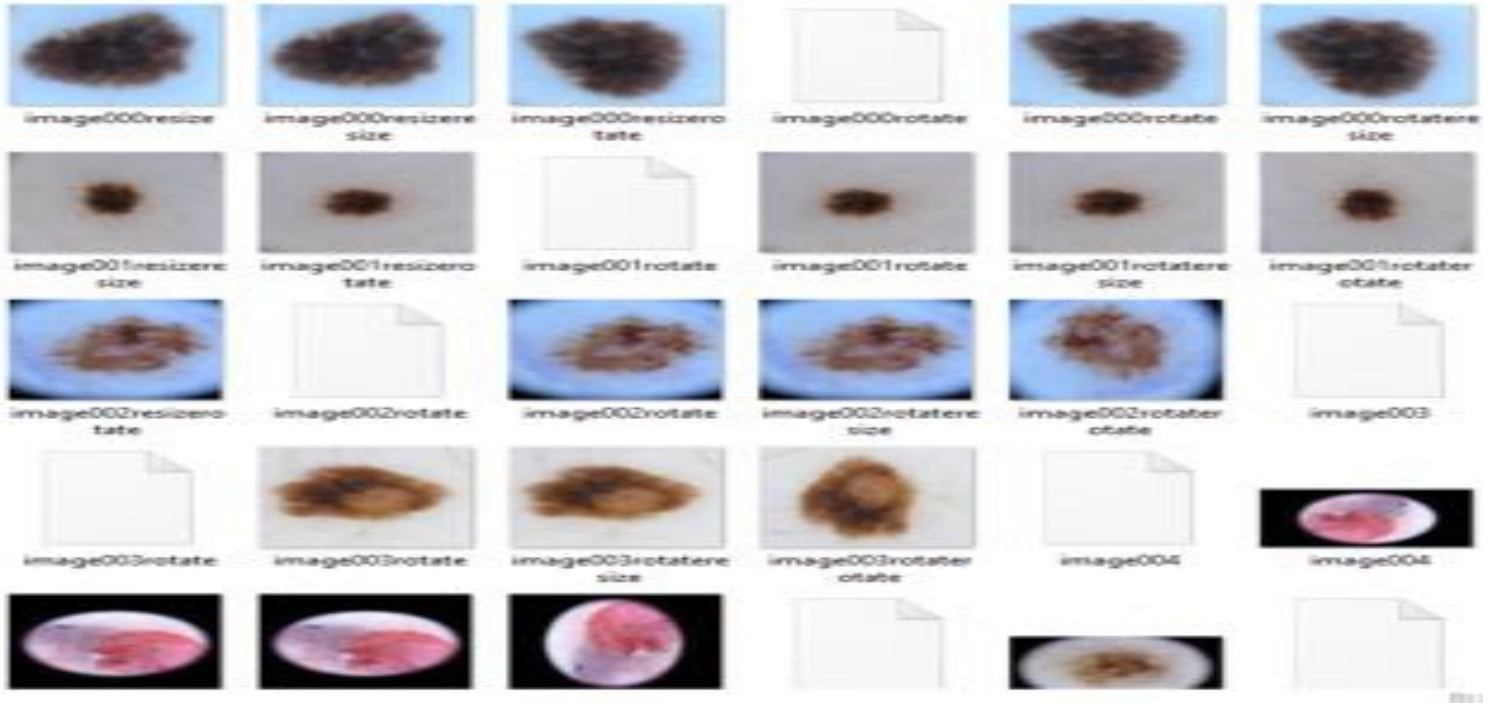
- Accessing to 300 images (potentially ~14,000 images later) and associated data (pixels, metadata – age/ gender/ image dimension…) from ISIC Archive API (Applications Programming Interface)
- The International Skin Imaging Collaboration (ISIC): Melanoma Project is an academia and industry partnership designed to facilitate the application of digital skin imaging to help reduce melanoma mortality.
- API – like a menu in a restaurant. When you know what you want, the restaurant (API) will prepare and do it for you. You don't need to know how
- Using Python, Pycharm & HPCC (High Performance Computing Cluster)

Steps in Machine Learning

- Preprocessing data
- (Loading data)
- Training data
- Predicting data

Step 1: Preprocess data

- Fetching Images and Making Data Frame



Step 1: Preprocess data

```
data_df
```

	Breslow	Clark	age_approx	anatom_site_general	benign_malignant	diagnosis	diagnosis_confirm_type	id1	localization	melanocytic	num	sex	site
0	NaN	NaN	55.0	anterior torso	benign	dysplastic nevus	None	1	Abdomen	True	0	female	bar
1	NaN	NaN	30.0	anterior torso	benign	dysplastic nevus	None	2	Abdomen	True	1	female	bar
2	NaN	NaN	60.0	upper extremity	malignant	Melanoma in situ	histopathology	3	Upper limb	True	2	female	bar
3	NaN	NaN	30.0	upper extremity	benign	dysplstic nevus	None	4	Upper limb	True	3	male	bar
4	NaN	NaN	80.0	posterior torso	malignant	Melanoma	histopathology	5	Back	True	4	male	bar
5	NaN	NaN	40.0	anterior torso	benign	Nevus	None	6	Abdomen	True	5	female	bar
6	NaN	NaN	25.0	posterior torso	benign	dysplastic nevus	None	7	Back	True	6	female	bar
7	NaN	NaN	25.0	posterior torso	benign	dysplastic	None	8	Back	True	7	female	bar

Step 1: Preprocess data

- Preprocess data

  Labeling each image with the name and 0 (benign) or 1 (malignant)

```
image_labels_2 = []

for index, row in data_df.iterrows():
    filename = "image%03d.jpg" % index
    malignant = int("malignant" in row["benign_malignant"])
    benign = int("benign" in row["benign_malignant"])
    image_labels_2.append([filename, malignant])
```

```
print(image_labels_2)
```

```
[['image000.jpg', 0], ['image001.jpg', 0], ['image002.jpg', 1], ['image003.jpg', 0], ['image004.jpg', 1], ['image005.jpg', 0],
['image006.jpg', 0], ['image007.jpg', 0], ['image008.jpg', 0], ['image009.jpg', 0], ['image010.jpg', 0], ['image011.jpg', 0],
['image012.jpg', 0], ['image013.jpg', 1], ['image014.jpg', 0], ['image015.jpg', 0], ['image016.jpg', 0], ['image017.jpg', 0],
['image018.jpg', 0], ['image019.jpg', 0], ['image020.jpg', 0], ['image021.jpg', 0], ['image022.jpg', 1], ['image023.jpg', 0],
['image024.jpg', 0], ['image025.jpg', 0], ['image026.jpg', 1], ['image027.jpg', 0], ['image028.jpg', 0], ['image029.jpg', 1],
['image030.jpg', 1], ['image031.jpg', 1], ['image032.jpg', 0], ['image033.jpg', 0], ['image034.jpg', 0], ['image035.jpg', 1],
['image036.jpg', 1], ['image037.jpg', 0], ['image038.jpg', 0], ['image039.jpg', 0], ['image040.jpg', 1], ['image041.jpg', 0],
['image042.jpg', 0], ['image043.jpg', 1], ['image044.jpg', 0], ['image045.jpg', 0], ['image046.jpg', 1], ['image047.jpg', 0],
['image048.jpg', 0], ['image049.jpg', 1], ['image050.jpg', 0], ['image051.jpg', 0], ['image052.jpg', 0], ['image053.jpg', 0],
['image054.jpg', 1], ['image055.jpg', 0], ['image056.jpg', 1], ['image057.jpg', 0], ['image058.jpg', 0], ['image059.jpg', 0],
['image060.jpg', 0], ['image061.jpg', 0], ['image062.jpg', 0], ['image063.jpg', 0], ['image064.jpg', 0], ['image065.jpg', 0],
['image066.jpg', 0], ['image067.jpg', 0], ['image068.jpg', 0], ['image069.jpg', 0], ['image070.jpg', 1], ['image071.jpg', 0],
['image072.jpg', 0], ['image073.jpg', 0], ['image074.jpg', 1], ['image075.jpg', 0], ['image076.jpg', 1], ['image077.jpg', 1],
['image078.jpg', 1], ['image079.jpg', 0], ['image080.jpg', 0], ['image081.jpg', 0], ['image082.jpg', 0], ['image083.jpg', 0],
['image084.jpg', 0], ['image085.jpg', 0], ['image086.jpg', 0], ['image087.jpg', 0], ['image088.jpg', 0], ['image089.jpg', 0],
['image090.jpg', 0], ['image091.jpg', 0], ['image092.jpg', 0], ['image093.jpg', 0], ['image094.jpg', 0], ['image095.jpg', 0],
['image096.jpg', 0], ['image097.jpg', 0], ['image098.jpg', 0], ['image099.jpg', 0], ['image100.jpg', 0], ['image101.jpg', 0],
```

Step 2: Loading data

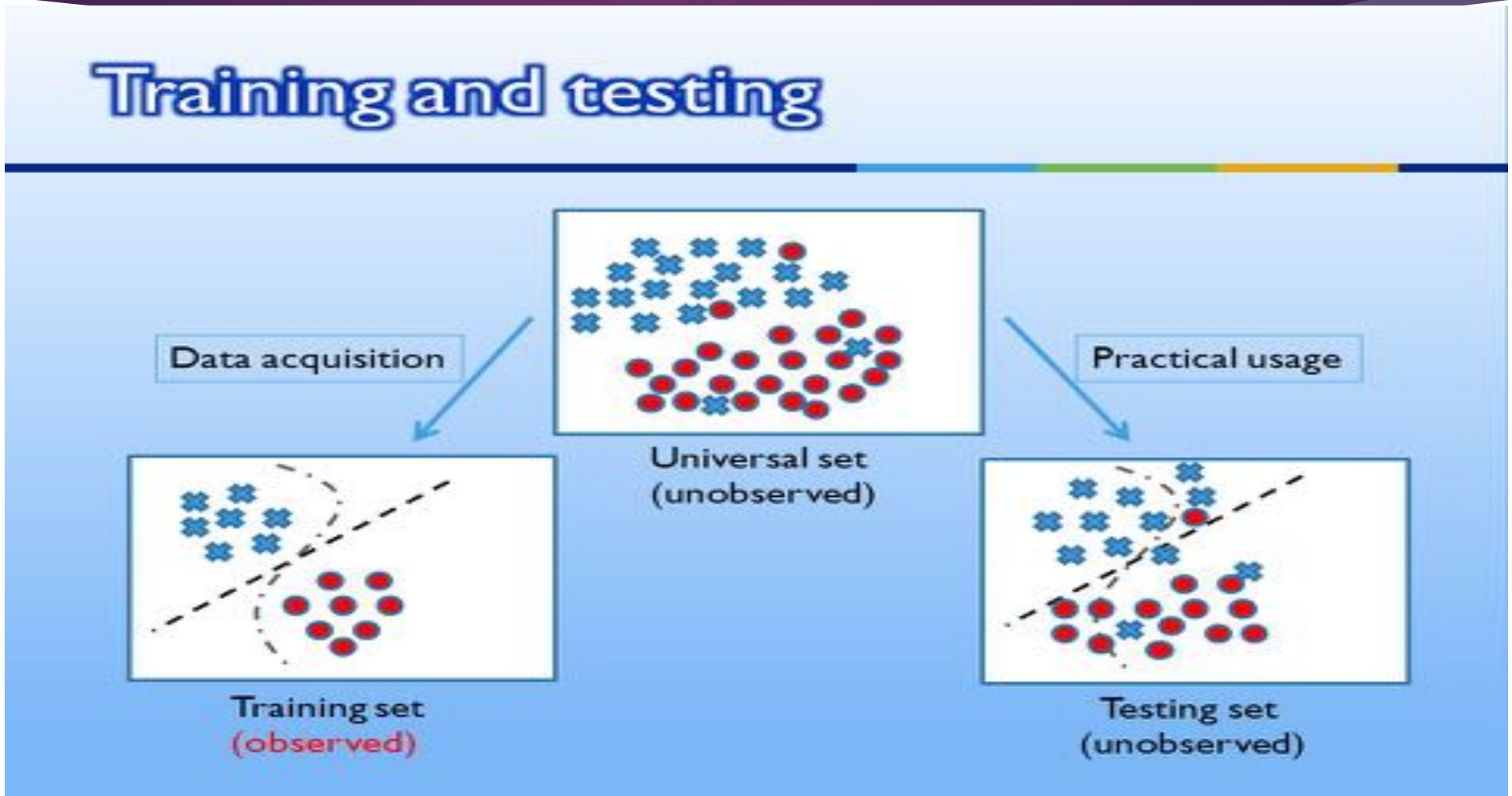
- Loading data as arrays

```
In [60]: X_colcorr[3]

Out[60]: array([[[0.66808279, 0.73850763, 0.78278867],
                 [0.71535948, 0.78578431, 0.83022876],
                 [0.7206427 , 0.79106754, 0.83551198],
                 ...,
                 [0.70446623, 0.77881264, 0.8370915 ],
                 [0.70816993, 0.78251634, 0.83594771],
                 [0.66421569, 0.73856209, 0.78665577]],

                [[0.64607843, 0.71650327, 0.76094771],
                 [0.68169935, 0.75212418, 0.79656863],
                 [0.68088235, 0.75130719, 0.79575163],
                 ...,
                 [0.67385621, 0.74820261, 0.80441176],
                 [0.66895425, 0.74330065, 0.79558824],
                 [0.62303922, 0.69738562, 0.74558824]],

                [[0.65915033, 0.72957516, 0.77396514],
                 [0.69133987, 0.76176471, 0.80620915],
                 [0.68300654, 0.75343137, 0.79787582],
                 ...,
                 [0.66786492, 0.74221133, 0.8004902 ],
                 [0.67042484, 0.74477124, 0.8004902 ],
                 [0.63589325, 0.708878  , 0.76083878]],

                ...,

                [[0.67058824, 0.74488017, 0.7956427 ],
                 [0.70359477, 0.77794118, 0.83022876],
```

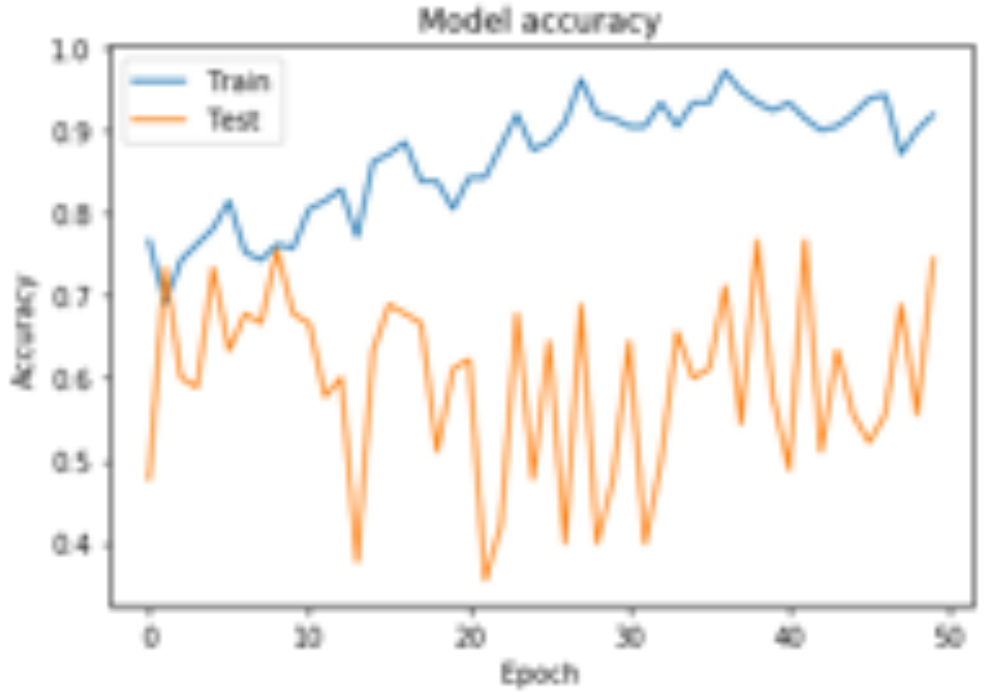
Step 3: Training data

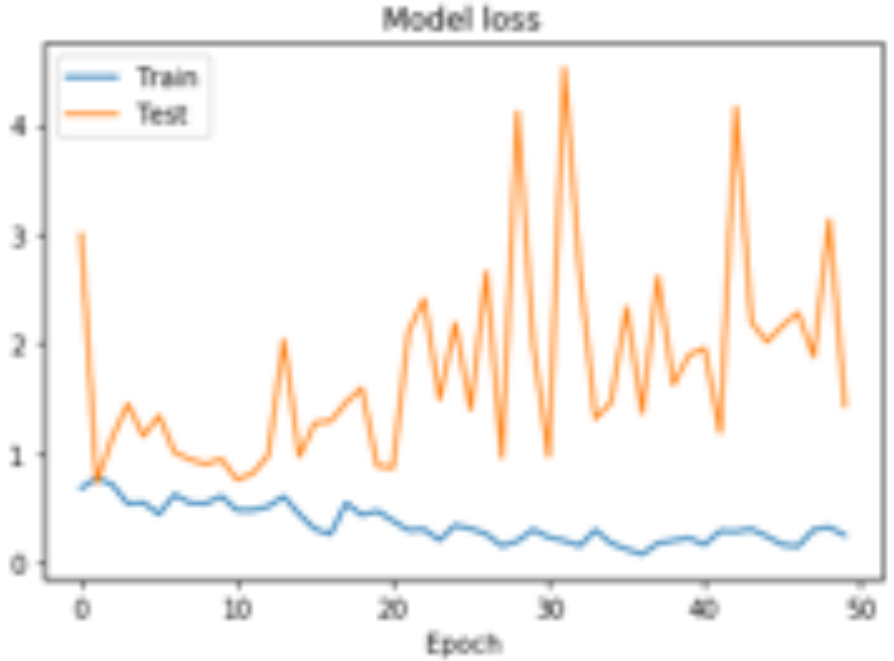
- Choosing deep neural network – **Resnet50** (Residual Network): it allowed us to train extremely deep neural networks with 150+layers successfully.
- We would want to use **Random Kitchen Sink** Algorithm to compare the efficiency

Training Set and Testing Set



Step 3: Training data





Lessons Learned

- How to google and read information (seriously Victoria?)
- Embrace ambiguity
- Work as a team and independent
- Python, Machine learning, deep learning and AI

Acknowledgement

- Dr. Hamsa Bastani - mentor
- Arielle Anderer – PhD student
- Joanne Levy, staff and friends in SUMR

Questions and Comments