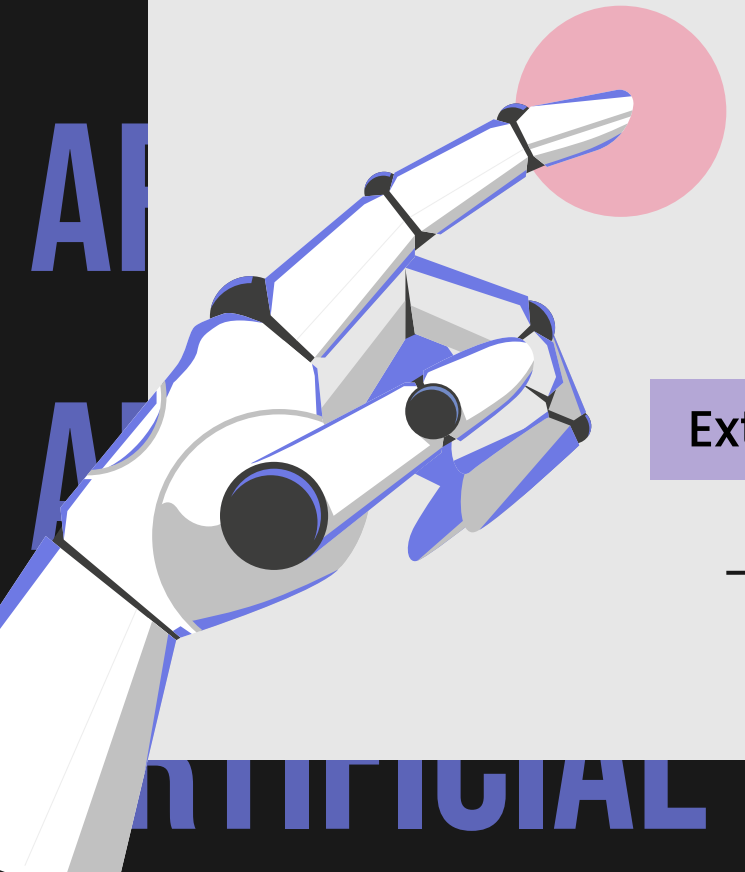


AIR CURE

TEAM JUNIOR GHOSTS

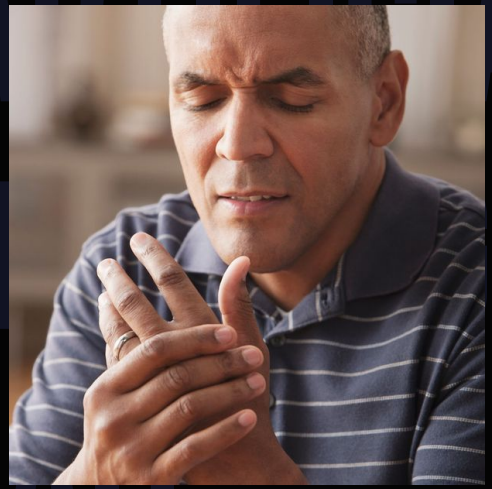
Extremely Low Cost Gamified Therapy

Mohammed // Fazith // Raja

PROBLEM

Physiotherapy Wellness Centers are

- Not Affordable enough
- Not Engaging enough
- Not Well Known enough



1 in 4 Adults will suffer from Arthritis in lifetime

That's 550 million people in SG!

PROBLEM

Physiotherapy Wellness Centers are

- Not Affordable enough
- Not Engaging enough
- Not Well Known enough

Medication and Therapy cost **SGD$130/Month**

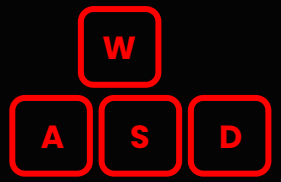
Physiotherapy gloves cost over 300 SGD

Solution

A Gamified Gesture Based Approach to actively promote physiotherapy for Arthritis and Carpal Tunnel

ARTIFICIAL INTELLIGENCE (AI)

Low Cost at **only SGD$1***

Ability to play ANY online game that uses 

keys

UNLIMITED GAMES || UNLIMITED FUN

(* Cost is derived from weight of 3D print used in glove)

TRAINING - *BEFORE IMPROVEMENTS*

Import TensorFlow and other libraries

```
In [2]: import matplotlib.pyplot as plt
        import numpy as np
        import os
        import PIL
        import tensorflow as tf
        import pathlib
        from tensorflow import keras
        from tensorflow.keras import layers
        from tensorflow.keras.models import Sequential
```

Check how many images in images folder

```
In [3]: image_data_dir = "./images"
        image_data_dir = pathlib.Path(image_data_dir)
        image_count = len(list(image_data_dir.glob('*/*.png')))
        image_count_palm = len(list(image_data_dir.glob('palm/*.png')))
        image_count_one = len(list(image_data_dir.glob('one/*.png')))
        image_count_two = len(list(image_data_dir.glob('two/*.png')))
        image_count_three = len(list(image_data_dir.glob('three/*.png')))
        image_count_four = len(list(image_data_dir.glob('four/*.png')))
        image_count_five = len(list(image_data_dir.glob('five/*.png')))
        image_count_c = len(list(image_data_dir.glob('c/*.png')))
        image_count_l = len(list(image_data_dir.glob('l/*.png')))
        image_count_ok = len(list(image_data_dir.glob('ok/*.png')))
        image_count_thumb = len(list(image_data_dir.glob('thumb/*.png')))
        print("Total = "+str(image_count))
        print("Palm = "+str(image_count_palm))
        print("One = "+str(image_count_one))
        print("Two = "+str(image_count_two))
        print("Three = "+str(image_count_three))
        print("Four = "+str(image_count_four))
        print("Five = "+str(image_count_five))
        print("C = "+str(image_count_c))
        print("L = "+str(image_count_l))
        print("ok = "+str(image_count_ok))
        print("thumb = "+str(image_count_thumb))

        Total = 3600
        Palm = 360
        One = 360
        Two = 360
        Three = 360
        Four = 360
        Five = 360
        C = 360
        L = 360
        ok = 360
        thumb = 360
```

CNN Sequential Model
3600 Total Pics

```
In [12]: AUTOTUNE = tf.data.AUTOTUNE

         train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size=AUTOTUNE)
         val_ds = val_ds.cache().prefetch(buffer_size=AUTOTUNE)
```

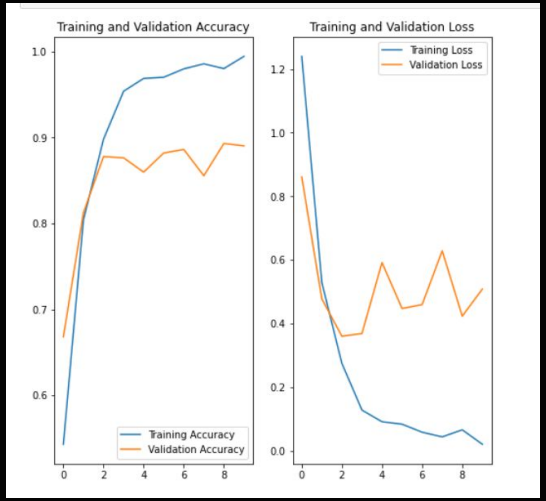
The RGB channel values are in the [0, 255] range. This is not ideal for a neural network; in general we should seek to make your input values small. Here, we will standardize values to be in the [0, 1] range by using a Rescaling layer.

```
In [13]: normalization_layer = layers.experimental.preprocessing.Rescaling(1./255)

In [14]: normalized_ds = train_ds.map(lambda x, y: (normalization_layer(x), y))
         image_batch, labels_batch = next(iter(normalized_ds))
         first_image = image_batch[0]
         # Notice the pixels values are now in `[0,1]`.
         print(np.min(first_image), np.max(first_image))

         0.0005286801 1.0
```

Used cache to speed up image loading



Overfitting
Problem: Not enough Dataset

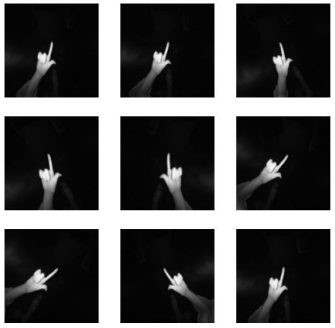
TRAINING - *AFTER IMPROVEMENTS*

Data augmentation

```
In [22]: data_augmentation = keras.Sequential(
           [
             layers.experimental.preprocessing.RandomFlip("horizontal",
                                                          input_shape=(img_height,
                                                                       img_width,
                                                                       3)),
             layers.experimental.preprocessing.RandomRotation(0.1),
             layers.experimental.preprocessing.RandomZoom(0.1),
           ]
         )
```

Visualize few augmented examples look like by applying data augmentation to the same image several times.

```
In [23]: plt.figure(figsize=(10, 10))
         for images, _ in train_ds.take(1):
           for i in range(9):
             augmented_images = data_augmentation(images)
             ax = plt.subplot(3, 3, i + 1)
             plt.imshow(augmented_images[0].numpy().astype("uint8"))
             plt.axis("off")
```



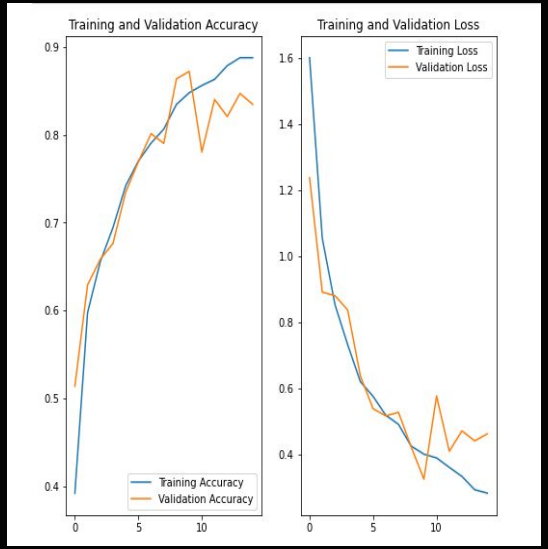
Data Augmentation to generate additional Dataset

Dropout and retrain

Apply Dropout to a layer it randomly drops out (by setting the activation to zero) a number of output units from the layer during the training process. Dropout takes a fractional number as its input value, in the form such as 0.1, 0.2, 0.4, etc. This means dropping out 10%, 20% or 40% of the output units randomly from the applied layer. Create a new neural network using layers.Dropout, then train it using augmented images.

```
In [24]: model = Sequential([
           data_augmentation,
           layers.experimental.preprocessing.Rescaling(1./255),
           layers.Conv2D(16, 3, padding='same', activation='relu'),
           layers.MaxPooling2D(),
           layers.Conv2D(32, 3, padding='same', activation='relu'),
           layers.MaxPooling2D(),
           layers.Conv2D(64, 3, padding='same', activation='relu'),
           layers.MaxPooling2D(),
           layers.Dropout(0.2),
           layers.Flatten(),
           layers.Dense(128, activation='relu'),
           layers.Dense(num_classes)
         ])
```

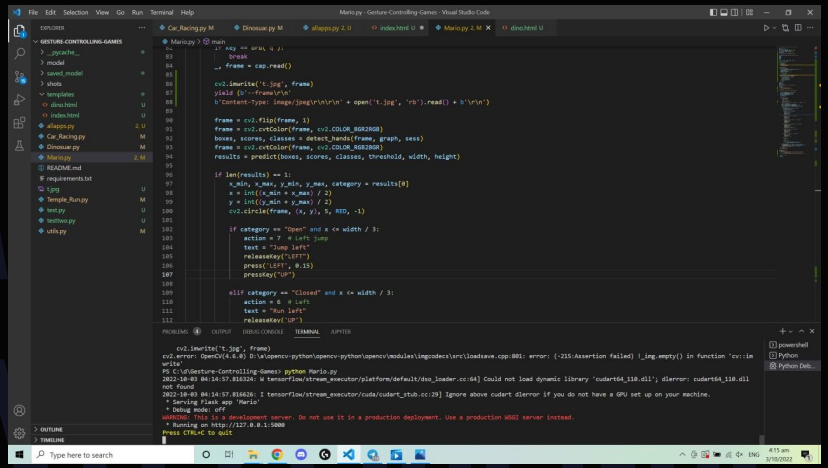
Implemented Dropout regularization for nodes



Nice Fitted Model

Train accuracy: 89.6%
Test accuracy: 83.47%

OUR PROJECT - GAMIFIED GESTURE DETECTION

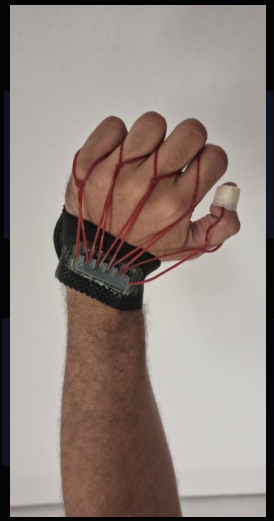


Mapped prediction of gestures to
Keyboard W A S D
using PyKey and Press libraries

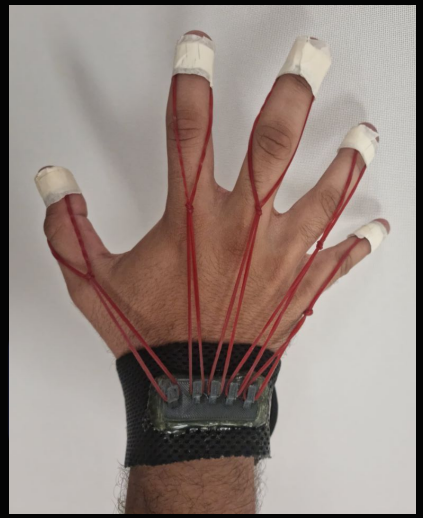
Used Flask web development and
OpenCV to demonstrate running desired
game *(in this case: Super Mario)*

OUR PROJECT - 3D PRINTED PHYSIOTHERAPY GLOVE

"A"
(Go Left)



"D"
(Go Right)



"W"
(Go Straight/Jump)

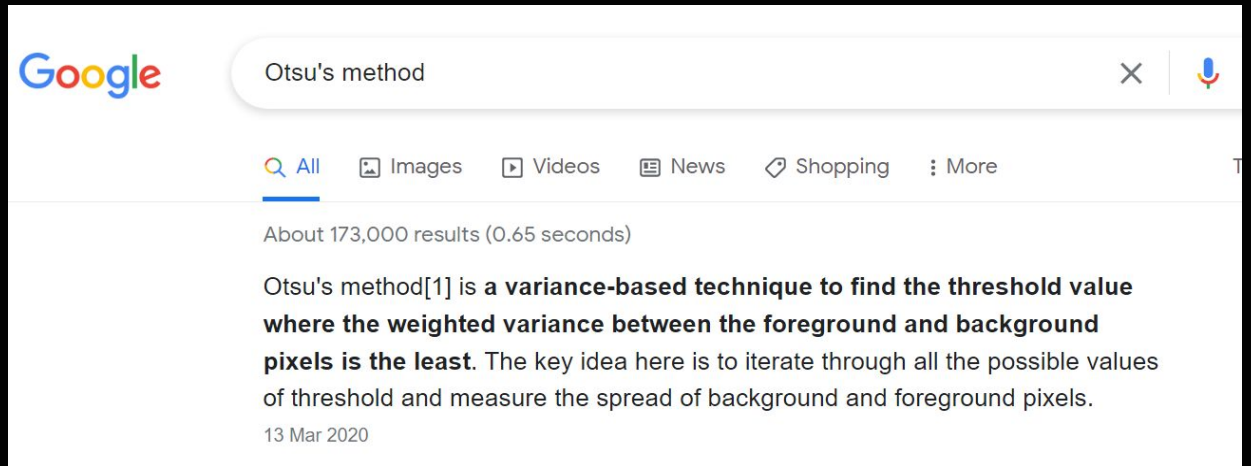
Another orientation to train upper hand muscles

FUTURE DEVELOPMENT - BETTER AI DETECTION MODEL

Due to Time Constraints, We weren't able to completely fine tune AI Model

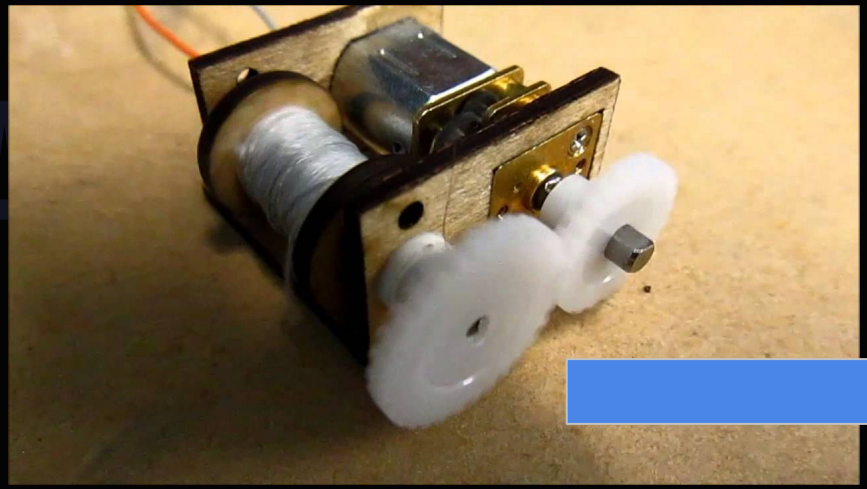
```
In [15]: model.compile(optimizer='adam',
                       loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
                       metrics=['accuracy'])
```

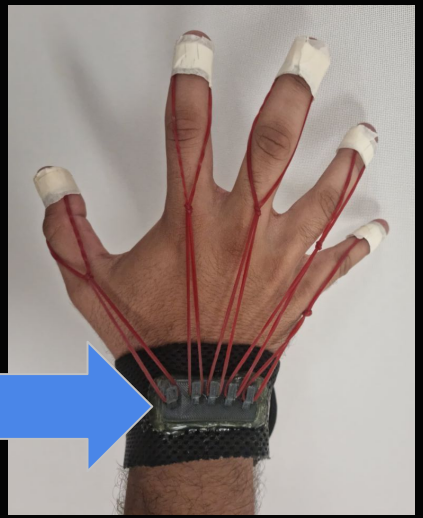
(RIGHT NOW)



(IN FUTURE)

FUTURE DEVELOPMENTS - HARDWARE





3D Printed Winches on each end of the bands for personalised adjustable tension

FUTURE DEVELOPMENTS - ASSISTIVE TECH FOR STUDENTS

- We can also propel our product towards learning for students with disabilities
- Allows them to learn/ do assignments through assistive technology provided by AIRCure
- Eliminates Teaching Assistant writing/doing things the student asks them to write

OUR PROJECT - BUSINESS/COMMERCIAL VIABILITY

1) Partner Local Physiotherapy Clinics

2) 1$ affordable therapy gloves

3) Access to MILLIONS of online games

SUPREMELY VERSATILE TEAM

SP Singapore Polytechnic

Mohammed HK
Diploma in Computer Engineering

Fazith Ismail
Diploma in Computer Engineering

Raja
Diploma in Computer Engineering

Engaging therapy for thousands of people in Singapore