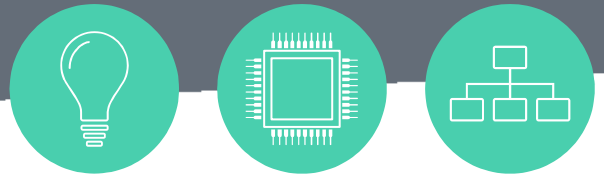
Team: Soorya Narayanan & Zhitao Li

GaitDisabilityML



Based On

Automatic recognition of gait related health problems in the elderly (people) using machine learning

Bogdan Pogorelc & Zoran Bosnić & Matjaž Gams

Multimed Tools Appl (2012) 58:333–354
DOI 10.1007/s11042-011-0786-1

Outline

- Introduction
- Methodology
- Result
- Conclusion & Further Work

Introduction

- Some diseases can manifest in patients’ gait.
- Develop a motion capture system without the use of IR and Retroreflective tags.
- Optimize feature selection to match accuracy.
- Develop k-NN and NN algorithms that will predict presence of ailments like, Parkinsons, Hemiplegia and others.

Data Collection

Ideal Camera Angle :

- 45 degree from front perspective (As shown)

4 Participants imitating 2 disabilities, and normal walk

- Total of 33 Samples:
    - 11 Normal
    - 11 Parkinson's
    - 11 Stroke

Sample of stroke patients' gait

Participants: Zhitao, Ziqi, Vidhu, Dennis.

Motion Capture

Video Recording :

- Mobile Camera : Moto 5G / Huawei Mate 40 back camera
- FPS: 30
- Resolution : 1K

Mediapipe :

- Compared to Tf_lite Lightning/ Thunder, there is better feedback
- Does Not rely on GPU acceleration
- However as CPU power drops, fps falls
    - With battery 22<fps<30
    - Without battery 16<fps<20
- 21 points tracked and extracted for skeleton embedding and Pose3D

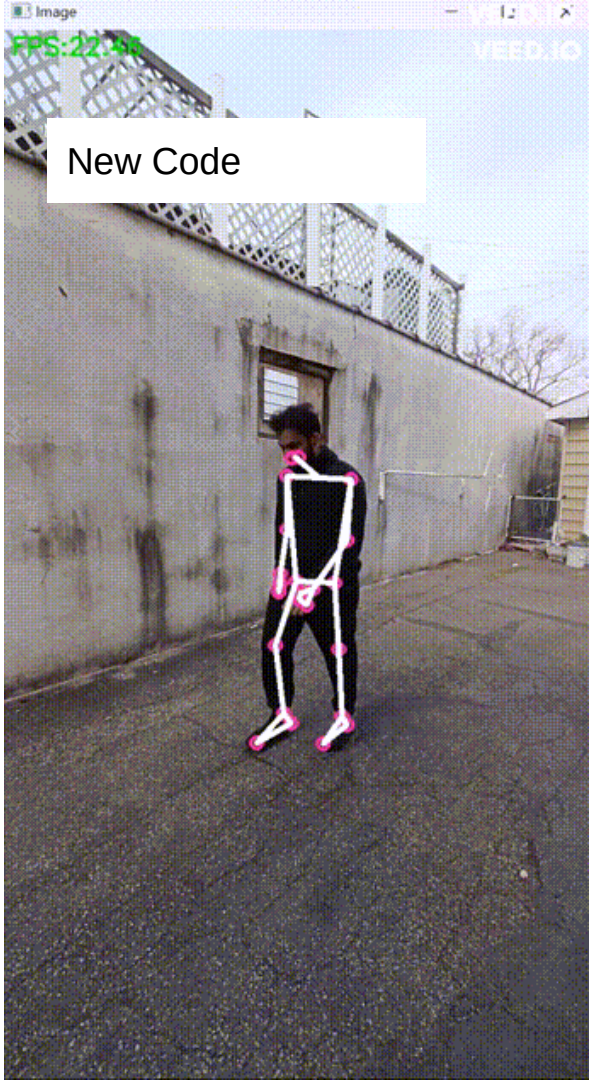


Image
FPS:22.46
VEED.IO
New Code

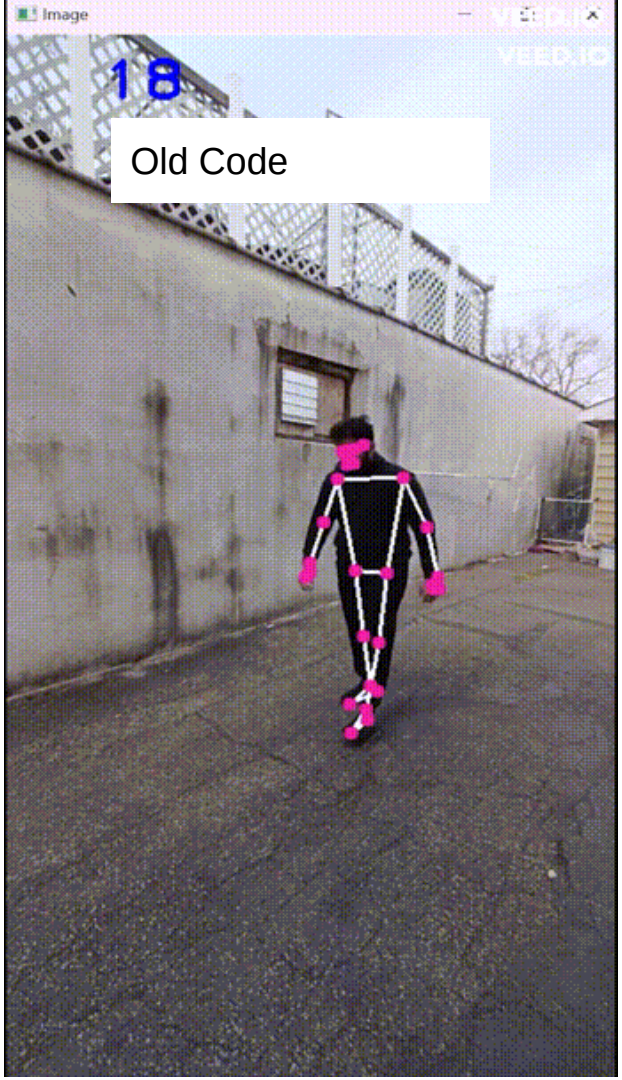


Image
18
VEED.IO
Old Code

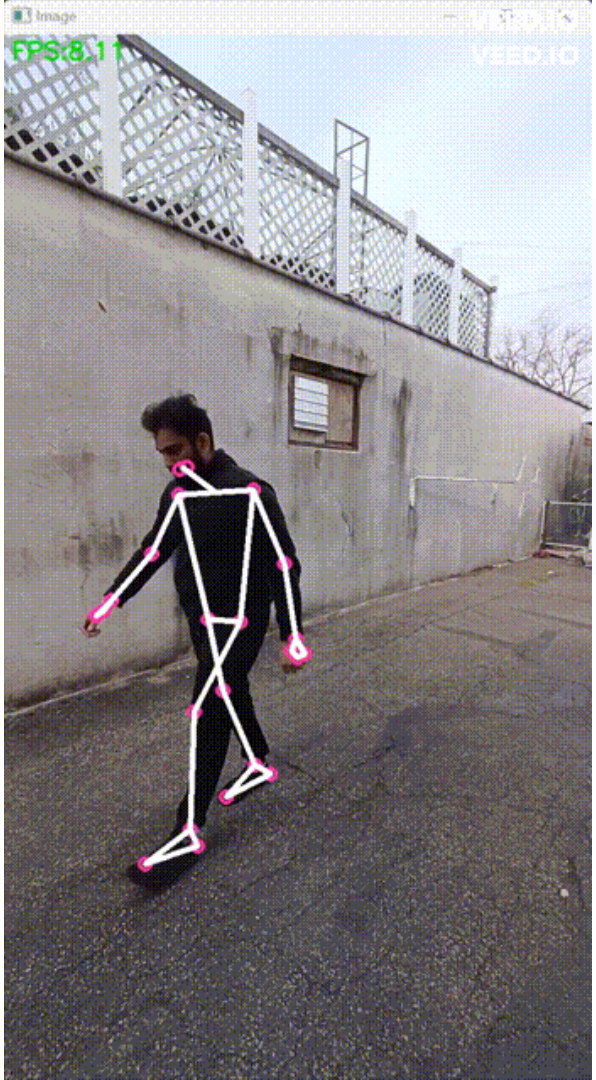
```
import copy
import pandas as pd
import cv2
import mediapipe as mp
import time
import math
import numpy as np
from matplotlib import pyplot as plt
from utils34 import CvFpsCalc
import plotly as ply
import csv
import F_C
import TensorPM as tm

fig = plt.figure()
axis3d = fig.add_subplot(111, projection="3d")
fig.subplots_adjust(left=0.0, right=1, bottom=0, top=1)
Tdata = []
testdata = []
timelist = []


class pDetect():
    # creating class for constructor decalration
    def __init__(self, mode=False, mcomplex=1, smooth=True, eseg=False, sseg=True, dconf=0.5, tconf=0.5):
        self.mode = mode
        self.mcomplex = mcomplex
        self.smooth = smooth
        self.eseg = eseg
        self.sseg = sseg
        self.dconf = dconf
        self.tconf = tconf
        self.mpPose = mp.solutions.pose
        self.mpDraw = mp.solutions.drawing_utils
        self.pose = self.mpPose.Pose(self.mode, self.mcomplex, self.smooth, self.eseg, self.sseg, self.dconf,
                                     self.tconf)

    def findpose(self, rs, vis):...
```

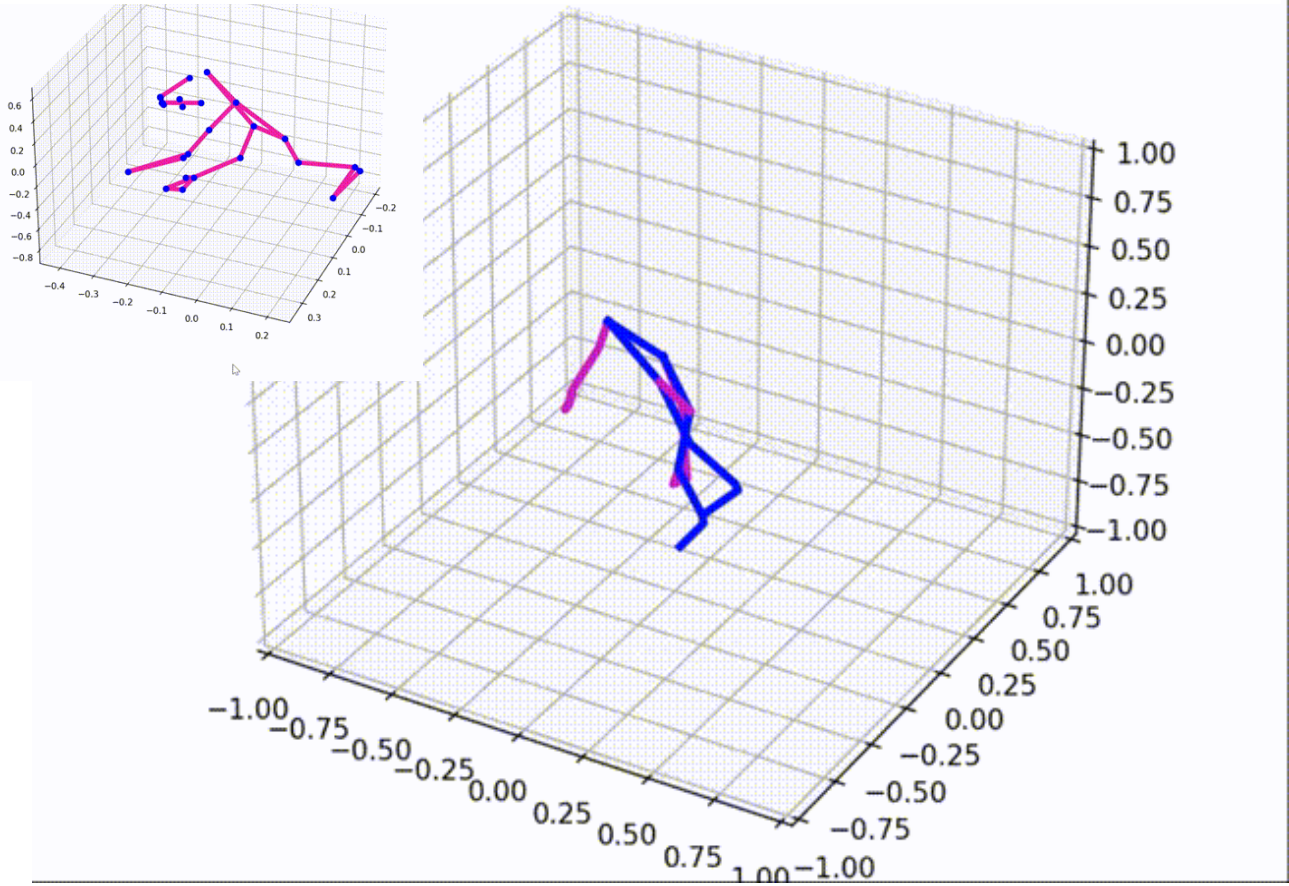
VEED.IO

Changes made:

- Disconnected Pose3D
- Split joint detection and connections
- Manually made the connections based on _.vis> vis_thresh
- Made a debug image to correct resolution

Results from Pose3D





```
import numpy as np

def featureBuild(mydata, rtime):
    Feature_block=[]
    tag_0 = mydata['0'].tolist();
    tag_0 = [eval(i) for i in tag_0]
    tag_1 = mydata['1'].tolist();
    tag_1 = [eval(i) for i in tag_1]
    tag_2 = mydata['2'].tolist();
    tag_2 = [eval(i) for i in tag_2]
    tag_3 = mydata['3'].tolist();
    tag_3 = [eval(i) for i in tag_3]
    tag_4 = mydata['4'].tolist();
    tag_4 = [eval(i) for i in tag_4]
    tag_5 = mydata['5'].tolist();
    tag_5 = [eval(i) for i in tag_5]
    tag_6 = mydata['6'].tolist();
    tag_6 = [eval(i) for i in tag_6]
    tag_7 = mydata['7'].tolist();
    tag_7 = [eval(i) for i in tag_7]
    tag_8 = mydata['8'].tolist();
    tag_8 = [eval(i) for i in tag_8]
    tag_9 = mydata['9'].tolist();
    tag_9 = [eval(i) for i in tag_9]
    tag_10 = mydata['10'].tolist();
    tag_10 = [eval(i) for i in tag_10]
    tag_11 = mydata['11'].tolist();
    tag_11 = [eval(i) for i in tag_11]
    tag_12 = mydata['12'].tolist();
    tag_12 = [eval(i) for i in tag_12]
    tag_13 = mydata['13'].tolist();
    tag_13 = [eval(i) for i in tag_13]
    tag_14 = mydata['14'].tolist();
    tag_14 = [eval(i) for i in tag_14]
    tag_15 = mydata['15'].tolist();
    tag_15 = [eval(i) for i in tag_15]
    tag_16 = mydata['16'].tolist();
    tag_16 = [eval(i) for i in tag_16]
```

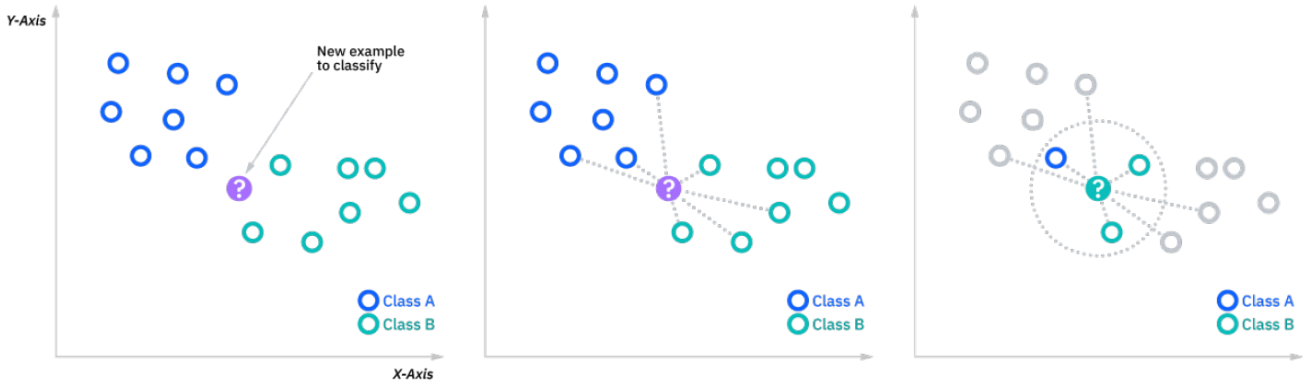
Feature Calculation

- 13 features calculated and imported for each sample

	A	B	C	D	E	F	G	H
1	0.265014	0.375776	0.105401	0.384921	0.229028	0.228486	0.333277	0.354807
2	158.217	160.2779	176.3164	150.298	125.4606	157.7938	157.1132	167.9722
3	0.992228	1.000018	1.000345	1.001033	1.00061	1.000847	0.99976	0.998535
4	61.82917	85.31326	13.51538	44.74679	30.73808	56.32007	79.46089	92.40748
5	0.273733	0.270727	0.286903	0.327293	0.303706	0.262557	0.318524	0.345949
6	0.350701	0.387562	0.163908	0.405888	0.311406	0.465662	0.393891	0.498928
7	0.897801	1.109228	0.747609	1.825083	1.124495	0.961714	1.429627	0.803635
8	0.637849	0.482083	0.020201	0.339592	1.427856	0.007604	2.078526	0.099677
9	0.26671	0.526551	0.237276	0.565506	0.305802	0.28323	0.442559	0.440184
10	0.598131	1.504097	0.684965	0.707951	0.931584	0.879312	0.51004	1.14057
11	0.034188	0.036364	0.052632	0.045802	0.061321	0.036735	0.045603	0.033333
12	26.73434	23.56269	31.97951	20.38284	7.789609	9.503015	24.15027	22.75685
13	-0.05805	0.079338	-0.19648	-0.01132	0.033331	0.04065	0.028353	0.000959

- 33 samples:
  - 11 normal
  - 11 parkinson's
  - 11 stroke
- Data sample:[13x33]
- Single training example: [13x1]

K - Nearest Neighbour (KNN)



https://www.ibm.com/topics/knn

K - Nearest Neighbour Result (k=5)

```
28 instance: [0, 3, 2]
29 instance: [5, 0, 0]
30 instance: [0, 1, 4]
31 instance: [0, 4, 1]
32 instance: [0, 3, 2]
33 instance: [4, 0, 1]
Succeed rate = 0.8181818181818182
 27 instances are correct in a total number of 33
Log:
 [0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1]

Process finished with exit code 0
```

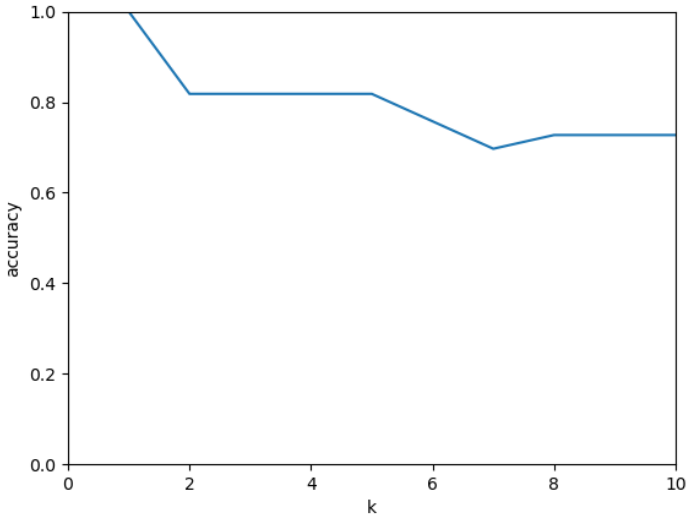
features	32	33
feature1	0.25058	0.388921
feature2	153.9213	151.968
feature3	0.999865	1.001663
feature4	32.28657	79.58781
feature5	0.357618	0.364977
feature6	0.47797	0.411431
feature7	1.024964	1.627512
feature8	2.061188	0.966702
feature9	0.298271	0.535502
feature10	0.860447	1.066748
feature11	0.046296	0.037037
feature12	7.386475	22.50145
feature13	-0.00986	-0.03563
label	3	1

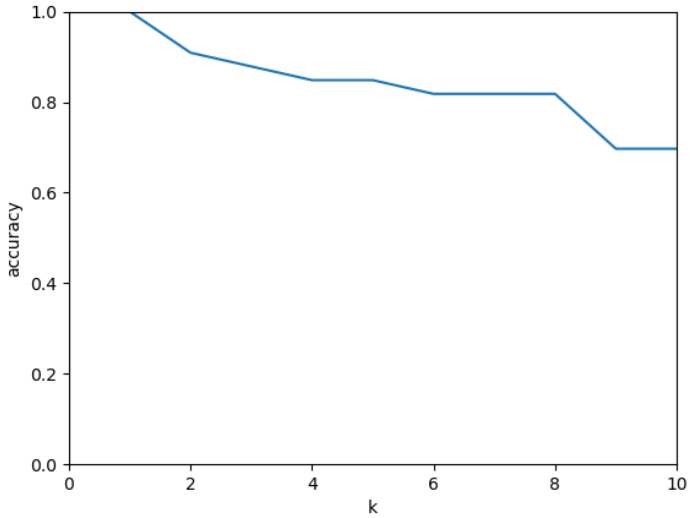
1 - normal
2 - parkinson
3 - stroke

The features' names were mentioned in previous presentation and are too long to show. It can be check in the paper [7].

K - Nearest Neighbour (KNN) Result

- K represents the number of neighborhood



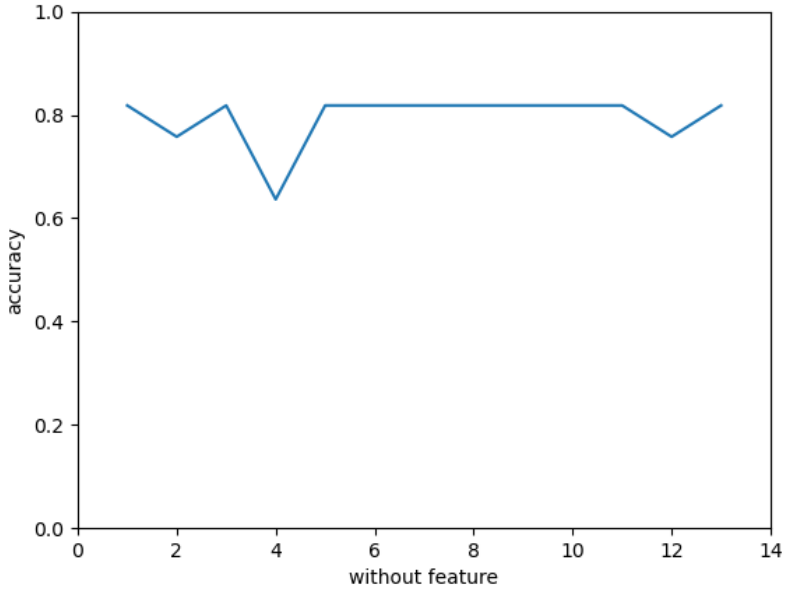


After Standardization

Before Standardization

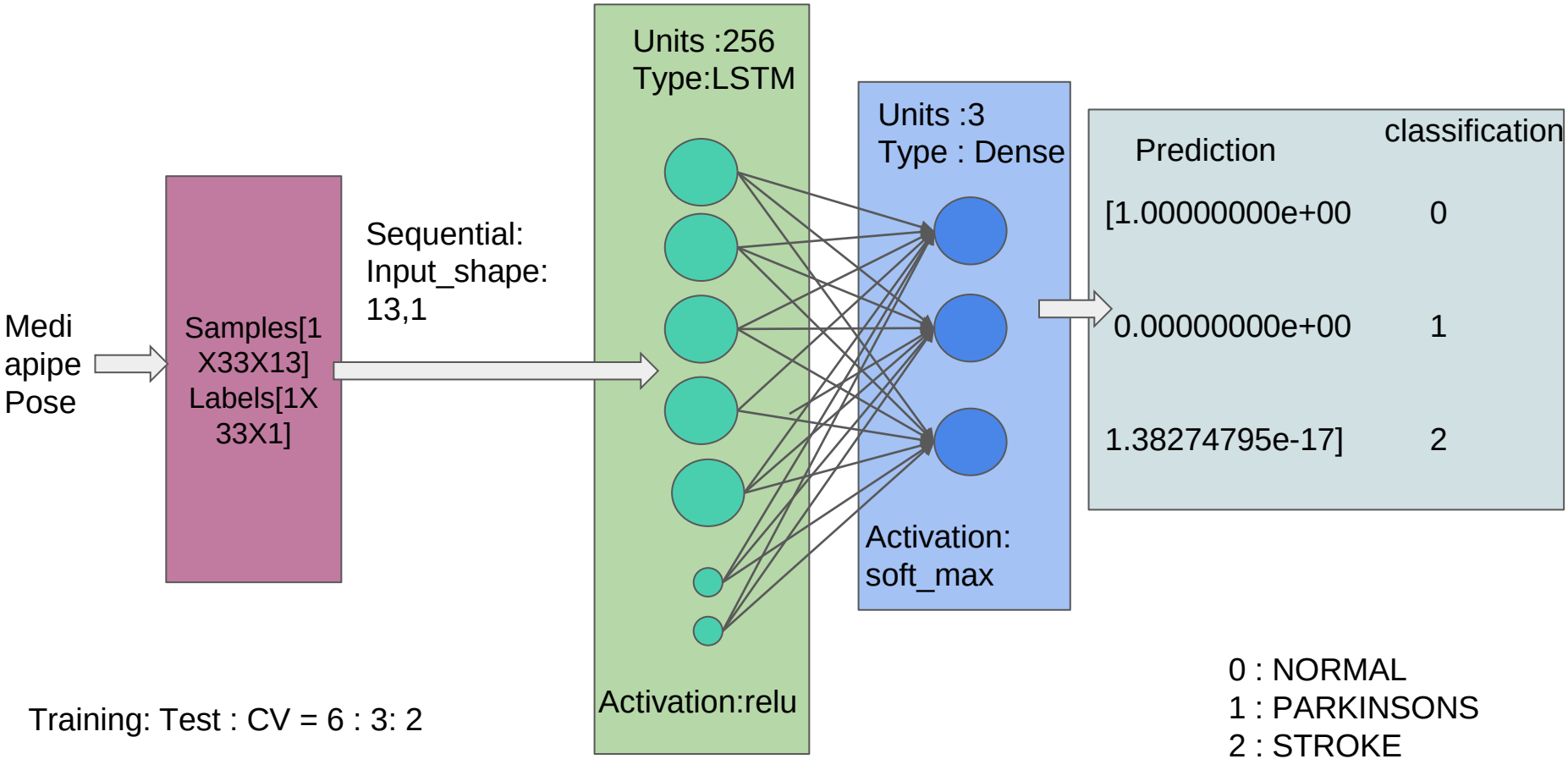
Standardization, often referred to as z-score Normalization, occasionally is a method for rescaling the values that meet the characteristics of the standard normal distribution while being similar to normalizing[3].

K - Nearest Neighbour (KNN) Result

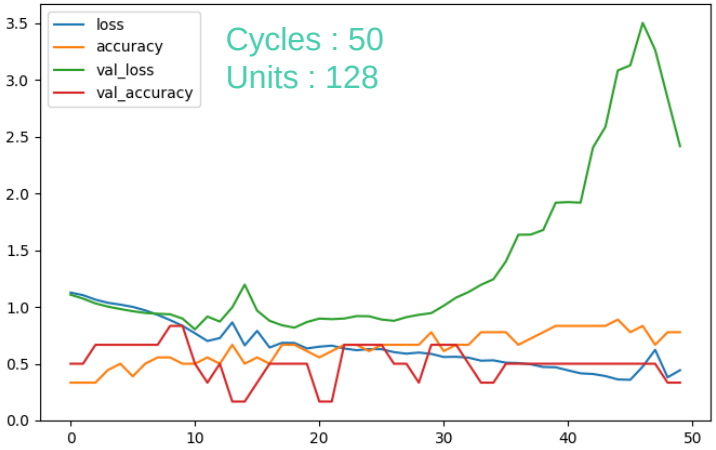


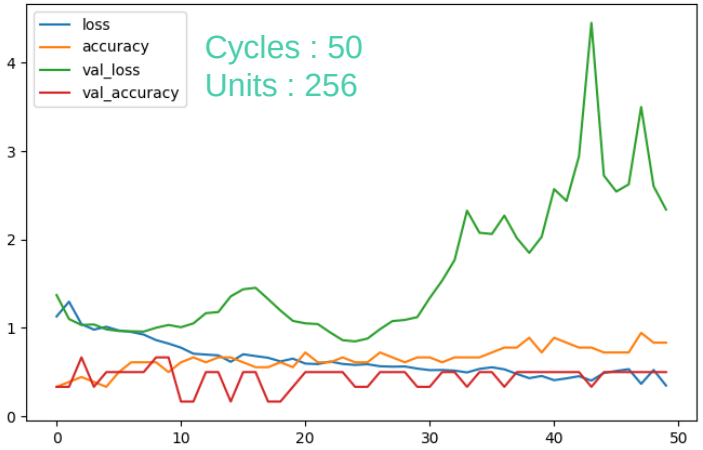
Accuracy without one of the features

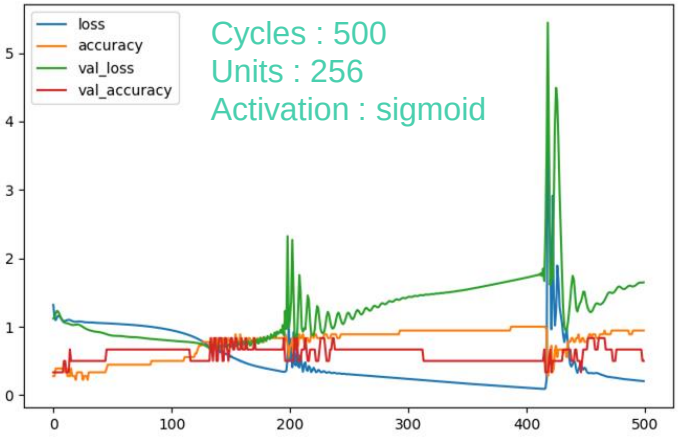
Neural Network Schematic

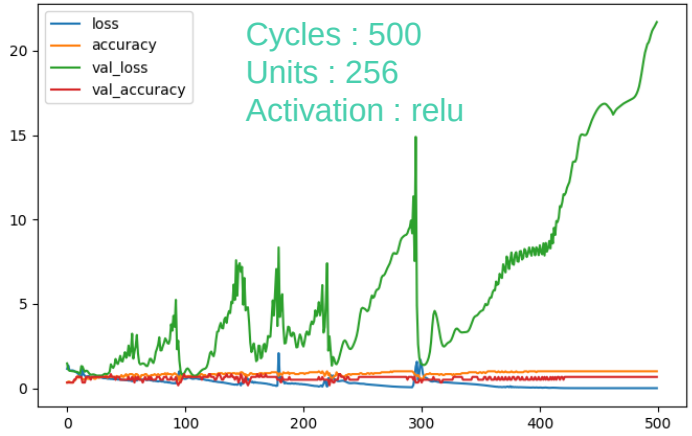


Neural Network (NN) Results

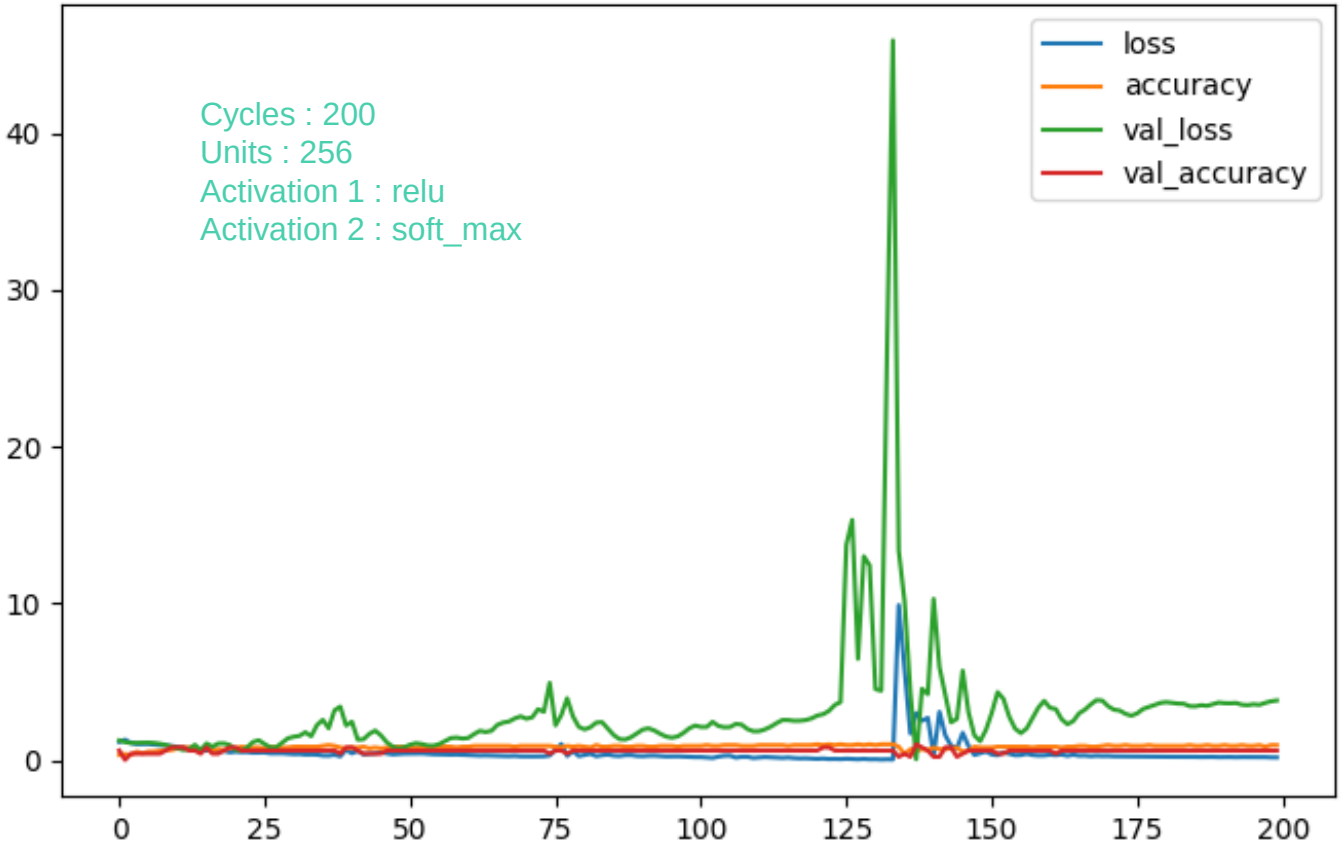








Best Run


Cycles : 200
Units : 256
Activation 1 : relu
Activation 2 : soft_max
loss
accuracy
val_loss
val_accuracy
40
30
20
10
0
0
25
50
75
100
125
150
175
200

[8]

Conclusion

- We have developed tools that can diagnosis some diseases from gait.
- We have done motion capture and classification.
- Accuracy of KNN: 75-82%, of NN: 92-96% training and 66-87% for CV

Timeline

Tutorials	Week -1	Week-2	Week-3	Week-4	Week-5	Week-6	Week-7	Week-8	Week-9	Week-10
Introduction to Machine Learning										
Basic Math and Gradient Descent I										
Gradient descent II and Regression										
Bias and Variance										
Introduction to NN										
NN										
Classification										
k-NN										
Intution into data selection										
Introduction to reinforcement learning										

Project Plan	Week -1	Week-2	Week-3	Week-4	Week-5	Week-6	Week-7	Week-8	Week-9	Week-10
Project Proposal										
Literature reivew										
Testing Setup										
Data Collection										
Feature Development										
K-NN Design		Lit Review	Lit Review							
Training K-NN										
Testing K-NN & Accuracy test										
NN Design		Lit Review	Lit Review							
Training NN										
Testing NN & Accuracy test										
Future Plans										

Appendix:1
Getting 3D pose from Mediapipe

Mediapipe:

- In python project,head over to python project interpreter add a new module/package named, mediapipe while making sure that the interpreter is selected as venv or virtual environment.
- Once environment and packages are installed restart becomes sometimes the path variables might need to be reset.
- Within the program, based on the use case do the following:
    - Create landmarks using the drawlandmarks function*
    - Develop connections using makeconnections function*
    - You may have to create a class object to self initialize the vectors
- BGR images have to converted to RGB and resolution has to be changed to allow for better landmark detection and training.
- On android devices, the camera has to be integrated to allow for real time capture.
- For 3D estimation extract 3d data based on relative data from world_landmarks
- Set projection as ‘111’ and plot using _.plot(). Create a pause to create motion

[* these are inbuilt functions within class mediapipe/pose and have be edited with care]

[** will be updated, system has to be developed for continuous capture]

Reference

[1] https://www.simplilearn.com/normalization-vs-standardization-article

[2] ABDUL SABOOR , TRIIN KASK , ALAR KUUSIK , (Member, IEEE), MUHAMMAD MAHTAB ALAM , (Senior Member, IEEE), YANNICK LE MOULLEC , (Member, IEEE), IMRAN KHAN NIAZI 2 (Senior Member, IEEE), AHMED ZOHA , (Member, IEEE), AND RIZWAN AHMAD , (Member, IEEE). *Latest Research Trends in Gait Analysis Using Wearable Sensors and Machine Learning: A Systematic Review,* IEEE Access 2020

[3] SOSHI SHIMADA,VLADISLAV GOLYANIK,WEIPENG XU,PATRICK PÉREZ,CHRISTIAN THEOBALT,*Neural Monocular 3D Human Motion Capture with Physical Awareness* ACM Trans. Graph., Vol. 40, No. 4, Article 83. Publication date: August 2021.

[4]https://developers.google.com/mediapipe/solutions/vision/pose_landmarker/

[5]https://tfhub.dev/google/lite-model/movenet/singlepose/lightning/3

[6]https://youtu.be/v1SoZ_S31pk : MSK Medicine

[7] Pogorelc, B., Bosnić, Z., & Gams, M. (2011). Automatic recognition of gait-related health problems in the elderly using machine learning. *Multimedia Tools and Applications*, *58*(2), 333–354. https://doi.org/10.1007/s11042-011-0786-1

THANK YOU