Report
Team: Cheetos
Sanidhia Maheshwari-2019MT10762
Raunak Jain-2019MT10719

Week 1:

- We tried our previous model of assignment 2.2 to classify the images. Initially we got accuracy of 35% with these models. We realized that our model was not able to generalize for different camera angles because of the different perspective of different camera angles.
- Then, we saw the testing and training images and we found that the testing images were a mirror image of one camera angle image of training data. So, we horizontally flipped all the testing images and got an accuracy of 41%. After analyzing randomly images from the training data, we decided to crop the image from left, right, top and bottom to remove the unnecessary background and ensured that the body in the image is not cropped. Before feeding the image datasets to the model, we resize the images to 64x64 so that model could run faster and can be run on our PC without facing memory errors.
- Then, we created a new training data set which includes the current training images and horizontally flipped training images(size of new training data is 2 times the original). We ran our assignment 2.2 model on that data and got an accuracy of 57%. (We have not altered testing data here).
- Then, we used random perspective with distortion scale = 0.6 and created new training data which includes original image, horizontally flipped image, random perspective images and

horizontally flipped with random perspective images(size of new training data is 4 times the original one). We trained the model using assignment 2.2 model and got accuracy of 67% (We have not altered testing data here).

- Then, we used random affine with rotation = (-30,30) and translation = (0.1,0.1) and created new training data which includes original image, horizontally flipped image, random perspective images, horizontally flipped with random perspective images, random affine images and horizontally flipped with random affine images (size of new training data is 6 times the original one). We trained the model using assignment 2.2 model and got accuracy of 74% (We have not altered testing data here).
- We have used these transformations as we found that using these transformations, training and testing images were becoming similar.
- Then, we thought that if we removed the background of the image, we might get better accuracy. So, we tried to remove the background of the images by first converting images to RGB and then using some conditions like if(red > 60,blue > 60,green > 60) then (255,255,255) otherwise (0,0,0). We tried to make the body of the person black and background white. Then, we trained our assignment 2.2 model on that image(with new training data) and got an accuracy of 71%. So, accuracy decreased using these images. We also made the background black and the body of the person white but got an accuracy of 73%.
- Then, we changed our model by increasing convolution layers blocks and fully connected layers and making dropout layers with p = 0.5 for all dropout layers. We also resize our images to (128,128) from (64,64). We trained our model using new training data and got accuracy slightly more than our previous best.

- We found using random affine was not working and so, we removed random affine images and used training data that includes original image, horizontally flipped image, random perspective images and horizontally flipped with random perspective images(size of new training data is 4 times the original one).We also applied random equalize and padding to all training images and testing images. Then, we ran our new model on new training data and got an accuracy of 75.5%.
- Then, we thought that there could be overfitting and so, we used early stopping. We found that using the number of epochs = 12 got our best accuracy of 75.782%.

0.7584424534803583
0.8287388008270159
0.8497587870434183
0.8656099241902137
0.875947622329428
0.8938662991040661
0.8973121984838043
0.8900758097863543
0.9076498966230186
0.9097174362508614
0.9286698828394211
0.9317711922811854
0.930737422467264
0.9310820124052378
0.9396967608545831
0.9362508614748449
0.9383184011026878
0.9424534803583735
0.9383184011026878
0.9383184011026878
0.9503790489317712
0.9514128187456926

Training Accuracy

1.5460125363778834
0.9107268760966905
0.7548068738743371
0.6689230280033513
0.613973487190464
0.5711229971226524
0.5380560269944515
0.5136824583342554
0.4915368302524382
0.47508025127006515
0.3864171555338829
0.3677502285232585
0.35631012609411106
0.34700657624349585
0.3390992387216173
0.33280733150758723
0.3264496061713526
0.3244011818641322
0.319337538610596
0.3120929552333466
0.2650050105787266
0.2516087825040278

Training Loss

Week 2:

- We tried to analyse the images that our model misclassified. When we manually saw the images, we found that most of the images classified are almost correct but there are many images that are classified as Tuladandasana but they didn't belong to Tuladandasana. Images were classified as Tuladandasana but actually they were Natarajasana. Then, we saw that Trikonasana and Parivrtta Trikonasana were also not classified properly. Same issue was there between Ardhachakrasana and Utkatasana.
- So, we thought that we first used our model that gave us the best accuracy and we created another model that is specialised in classifying the images of Tuladandasana and Natarajasana. So, whenever our model classifies images as Tuladandasana, we will use that model. For that we thought of using move-net.
- We used the movenet model of the tensorflow hub. We got 17 key points from that. We thought that camera angles are changing and it is rotation of images about the y-axis. So, if we know each point (x,y,z) coordinates in an image, then it's y coordinates will be the same in all the images(training and testing) and (x,z) coordinates will lie in a circle.
- So, we tried to find z coordinates of all the key points of the body. So, first we made the center of the body i.e. hips to (0.5,0.5,0) and translated all the points with respect to that. Then, for finding z-coordinates, we thought that the length of bones would be the same in all images, so we tried to find the length of bones of each body part. For that we saw images where a person is front facing as there the z-coordinate of all key points will be 0 and we will be able to find length of the body. We found the length of bones and so, we found the z-coordinate of each key point.

- Then, we knew that points x and z coordinates would lie in a circle, so we created new features i.e. x^2 and z^2. We run a neural network on that data to classify Tuladandasana and Natarajasana.
- We don’t get better accuracy using this. The main reason was we were keeping z-coordinates of key points to be positive but the relative position of leg and hands can be opposite which was in case of Tuladandasana and Natarajasana and so, our model doesn’t work.
- So, in week 2 we were not able to improve our accuracy.

Below is the image of training accuracy of different classes.

	Still	TriyakTada	Natavaras	Pranamasa	Santolana	Virabhadra	Tuladanda	Trikonasar	Natarajasa	Katichakra	Utkatasan	Vrikshasar	Ardhachak	Tadasana	ParivrittaT	Naukasana	Padahasta	Garudasar	Gorakshasana	Accuracy(Number of images that are correct/Total Number of images that are classified as that asan)
Still	1358	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0.994871795
TriyakTadasana	7	1545	0	0	1	1	2	0	0	25	0	2	16	51	0	0	5	0	0	0.933534743
Natavarasana	48	0	1494	2	4	0	0	0	3	8	0	13	4	0	0	0	0	22	2	0.93375
Pranamasana	35	3	13	1376	1	4	3	1	9	19	0	94	6	0	0	0	1	4	1	0.876433121
Santolanasana	3	0	0	0	1298	4	7	3	0	0	2	2	1	1	7	0	6	0	1	0.972284644
Virabhadrasana	4	0	0	1	10	1460	2	39	1	0	0	6	0	0	102	0	0	0	0	0.898461538
Tuladandasana	27	15	2	1	7	4	1279	2	0	15	10	23	156	12	1	0	19	6	1	0.809493671
Trikonasana	5	1	1	0	1	75	0	1351	0	1	0	4	0	0	159	2	0	0	0	0.844375
Natarajasana	43	3	2	26	5	6	26	1	1277	12	1	37	6	1	2	0	2	10	1	0.874058864
Katichakrasana	13	6	1	0	0	0	0	2	0	1474	0	7	2	0	0	0	0	0	0	0.979401993
Utkatasana	14	8	1	0	1	0	142	0	0	4	1331	2	149	5	0	0	6	0	2	0.799399399
Vrikshasana	18	4	3	86	3	25	9	0	8	18	3	1259	13	5	0	0	2	3	1	0.862328767
Ardhachakrasan	12	5	1	0	0	1	9	0	0	6	8	5	1483	4	0	0	1	0	0	0.966123779
Tadasana	10	11	0	1	0	0	4	0	0	12	4	9	68	1357	0	0	4	0	0	0.916891892
ParivrittaTrikona	4	0	0	0	1	44	1	45	0	1	0	3	0	0	1451	0	0	0	0	0.936129032
Naukasana	55	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1278	0	0	1	0.957303371
Padahastasana	17	6	0	3	1	0	21	0	0	10	0	2	92	6	0	0	1406	0	1	0.898402556
Garudasana	46	2	25	5	3	2	3	1	6	9	0	24	12	1	0	0	0	1406	0	0.910032362
Gorakshasana	13	2	1	0	2	0	2	0	0	6	3	3	0	0	0	0	1	1	1546	0.978481013
Accuracy(Number of images that are correct/Total Number of images that are that asan)	0.784065	0.959032	0.967617	0.916722	0.970105	0.897909	0.84702	0.934948	0.979294	0.909877	0.977239	0.84214	0.738546	0.940402	0.842136	0.998438	0.967653	0.96832	0.988491	

Week 3 and 4:

- We changed our model by decreasing the number of convolution layers and using the relu activation function instead of elu. But the accuracy hasn't improved.
- Then, we thought that we can use our movenet concepts by first classifying the camera angles and then finding the position of the z-coordinate for that camera angle. But we weren’t able to classify the camera angle correctly and so, our accuracy didn't improve.
- So, in week 3 and 4, we were not able to improve our accuracy.