Yog.ai

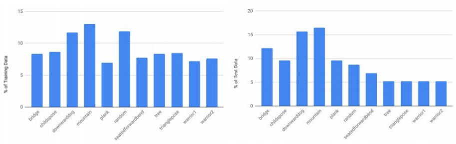
Anna Lai, Bhargav Reddy, Bruis van Vlijmen
{alai2, brkreddy, bvlijmen}@stanford.edu

Predicting

Pose estimation of humans from images is increasingly more developed and applicable to animation, choreography, VR, and more [1]. Many varieties of physical training have benefitted from A.I. applications. However, the field of yoga remains relatively unexplored. The goal of Yog.ai is to use pose recognition as a tool to allow for a person to practice different yoga poses and receive feedback on their form. This would be ideal for beginners to get a yoga practice started with minimal investment. This project covers the pose classification portion of this idea developing upon a baseline yoga-pose classification model from Marchenkova [2].

Data & Features

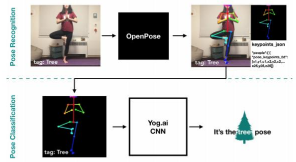
Our data set contains images of 10 common poses [2] with the addition of a random class of images of people [3].

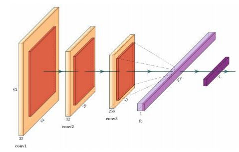


Using the OpenPose recognition tool [4], we extract key point features from the image data. The original data set of ~1,000 images (75% train, 25% test) are converted to skeleton images of connected key points. The processed training set is augmented to ~30,000 images then run through the Yog.ai CNN.

Model

Our approach which uses a RGB skeleton image run through a CNN classifier. We decide this since a CNN is less sensitive to the relative location of a figure in an image, compared to a fully connected NN, and we preserved the RGB images as the colors of the skeleton map to the limbs they represent.





With this architecture, we chose to tune learning rate, batch size, and number of dense nodes in the final layer given our relatively small dataset.

Results

We first began by tuning the learning rate and minibatch size using a randomized search of the space (all 100 epochs).

Learning Rate	Minibatch Size	Dense Nodes	Training Accuracy	Test Accuracy
0.0033	16	128	0.947	**0.712**
0.0001	64	128	0.925	0.668
0.0087	16	128	0.938	0.702
0.0063	64	128	0.918	0.645

After finding an optimal learning rate and minibatch size combination we tuned the number of dense nodes in the final layer.

Learning Rate	Conv Layers	Dense Nodes	Training Accuracy	Test Accuracy
0.0033	**3**	**256**	0.943	**0.778**
0.0033	3	80	0.943	0.725
0.0033	3	128	0.942	0.726
0.0033	2	128	0.946	0.707
0.0033	4	40	0.935	0.734

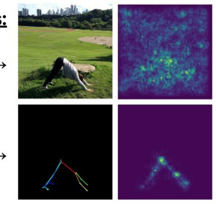
Discussion

Our hypothesis was that incorporating OpenPose as a 'preprocessing' step, would allow the network to train on the noise-free skeleton images. This proved to work well, as can be seen in the set of saliency maps shown below. We were able to reach a test-accuracy of 78%, compared to the SOA model (70%).

Saliency Maps:

Marchenkova Network →

Yog.ai Network →



However, we believe that we are be able to improve our results by expanding our dataset - as the small dataset made our network vulnerable to overfitting.

Future

For next steps we would incorporate more poses and train on a larger dataset and experiment with different architectures. We would also like to extend the functionality to providing the user feedback on their form. After the pose that the user is trying to do is classified, it would be compared with an ideal pose calibrated to the user to give advice and avoid injury.

References

[1] Toshev, A. & Szegedy, C (2014) Deeppose: Human pose estimation via deep neural networks.
[2] Marchenkova, A. (2018) Convolutional Neural Network for Classifying Yoga Poses. https://github.com/amarchenkova/yoga-pose-CNN/.
[3] Opelt, A., Fussenegger, M., Pinz, A., and Auer, P. (2004) Weak Hypotheses and Boosting for Generic Object Detection and Recognition.
[4] Cao, Z., Hidalgo, G., Simon, T., Wei, S.E. & Sheikh, Y. (2018) OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields.