Interactive Dance lessons through Human Body Pose Estimation and Skeletal Topographies Matching

Aayush Gupta, Ankit Arun, Shivangi Chaturvedi, Alpana Sharan, Suman Deb

Department of Computer Science and Engineering, NIT Agartala 799046, India

Abstract:

During elementary period of learning, children are enthusiastic and spontaneous. This is very intuitive and natural human behaviour. But, the growth of affordable mobile devices have given us the possibility to experience diverse learning without physical movement. It is a considerable threat towards their physical and mental development. In this work, we tried to explore the possibilities of natural human-movement interaction with machine through ExerLearning experience, that is, gamified learning through exercise. We device a learning game for teaching simple dance moves to children which encourages them to perform a pose to match the given dance move. The findings of this work are very encouraging for stimulating students to perform exercise while learning, which effectively shortens the learning time with increased enthusiasm toward the content. Though ExerLearning approaches have been in practice with special devices, our stress on work was cost effective skeletal movement design, independent of body markers and special devices. We attempt to replace conventional skeleton tracking, provided by devices such as Microsoft Kinect, with a single camera, leveraging the power of Deep Learning to perform the same task. The anecdotal description and quantitative data reveals the effect of the enhanced learning outcome on the participants. The design recommendations base on the experimental data can pave a new paradigm of interactive and cost effective ExerLearning.

Keywords: Pose Estimation, Deep Learning, Pose Matching

1. Introduction

The concept of Human Pose Estimation (HPE) opens up an exciting field of various new applications in the field of User Interaction, Assistive Technology, Gaming and many other fields. The harbinger of this technology, the Microsoft Kinect was a revolutionary device which is, even today, used extensively for various pose and depth sensing purposes. However, the hardware and software on which it was based, even for its newer v2 model, they are becoming outdated. Also, for most applications, the system is costly, harder to setup and sometimes requires additional hardware and software hacks to make it work. In this work, we attempt to employ a deep learning based approach for realtime 2D pose estimation. This makes sure that that our application will be accessible to anyone, anywhere as long as they have a camera. The pipeline of the application will require users to perform a sequence of dance moves for a song. The system will continuously evaluate the poses of the users and match them to a correct set of reference poses for scoring.

This motivation for developing this application was that fact that in nowadays busy life, we get much less time to relax and learn a new extraneous skill. Most people relax by watching movies and playing games. Although, it is good for the mind, our bodies also need some exercising. Thus, our application motivates users to take the initiative to learn something new, at the same time exercising themselves in the process. It is also useful for little children as a tool for making them learn dances and develop exercising habits which will benefit their overall health[1].

1.1. *Contributions*

In this paper we aim to demonstrate the ubiquitous nature of deep learning by exploring its utility in the task of pose estimation and the application of the acquired data in developing an interactive Digital Dance lesson application to teach user basic dance moves and exercises by monitoring their poses.

To enumerate, our contributions are twofold:

- Eliminating reliance on specialized hardware for 2D pose estimation from single monocular images.
- Using the estimated 2D human pose data to gain insights into developing effective human computer interaction design for teaching dance lessons and exercises to users.

In our work we use off the shelf Tensorflow[2] implementation of CMU-OpenPose[3] pose estimation model[4] based on[5,6]. For accurate analysis of pose data it is important that the pose estimation model be fairly accurate. Thus, we chose this model, which is based on the original caffe implementation by the same name and provides nearly state of the art accurate pose estimation with mAP of 70.6[5]. The Tensorflow implementation[4] was considered as it is easy to deploy it to other devices such as Android and Raspberry Pi. The model can be then packaged as an app for use in real time interactive applications.

2. Related Works

Human Pose Estimation has been traditionally used mainly in movie studios for producing special effects. Using markers placed on the body of the actors, the pose was estimated and a 3D mesh was overlaid on the footage so as to match the pose of the actor. Such special effects enabled filmmakers to change the body appearance of actors on the film and create realistic representations of animals and dinosaur through human actors. These pose estimation and matching techniques were also used in motion capture (MoCap) of actors which could then be animated for use in animated movies or computer games.

Over the time, such technology began to be used towards the consumer side through its applications in gesture based user interaction and gaming. Earlier such gesture based interactive applications in gaming and user interaction required the use of additional gear/controllers, either held by the user or fitted on to them. There were attempts to build a controller-free gear-free system requiring only vision to provide seamless, unhindered experience to the user. One of the very first successful attempts was the introduction of Microsoft Kinect. Started as Project Natal, the Kinect ecosystem was developed to provide the users means to control a gaming environment without the need for controllers. This made the experience for gaming immersive and intuitive as the users could then use the body movements to do practically anything. Later, the Kinect was adopted in many other fields from robotic navigation to gait recognition[9]. We thus discuss the related works in the fields of pose estimation and pose matching in gait assessment which were useful in developing our application.

Earlier pose estimation methods[7,8] involved using random decision forests and mean-shifting to regress and localize body parts from single depth images. This approach was used by Microsoft Kinect to infer joint positions from RGB and depth images captured by the Kinect camera and IR rangefinder[9]. With the advancements in high performance computing and parallelization in hardware, various deep learning techniques rose into prominence. They replaced traditional image processing and small scale machine learning techniques for visual recognition tasks. Using such sophisticated yet elegant algorithms and high performance computing hardware, it has been possible to perform such computational tasks nowadays that would have been almost impossible to do by any machine a few years ago. Such deep learning approaches are now being used in the development of very efficient and accurate human 2D pose estimation models which can infer the human pose from just RGB images without requiring any depth data.

Human 2D Pose Estimation using deep learning can be broadly categorized into top-down and bottom-up approaches. Top-down (Person-First) approaches[10,11] employ a person detector and perform single-person pose estimation for each detection leveraging existing single-person pose estimation techniques[6,12,13,14,15,16,17,18,19]. These top-down approaches suffer from early commitment where the failure of person detector causes the pose for the missed person be not detected. Runtime of these approaches is proportional to the number of people as the pose estimation is run for each person detection. On the other hand, bottom-up (Part-First) approaches [20,21,22] are attractive in that they offer robustness to early commitment and have runtimes invariant of the persons in the image. In this they first detect all body parts in image and then run a decoding algorithm to group the detections into person instances. However, the process of associating body parts involves integer linear programming calculations which are sometimes NP-Hard and take hours to solve. Some bottom-up approaches[22] directly regress additional offset vectors to make the association task computationally lighter.

In tasks such as Activity Recognition[23] and Gait Analysis[25,30], the pose data is usually matched to a labeled dataset of known activities, while in the latter the pose data of a gait cycle is matched to a database of authenticated pre-recorded gait cycles[24]. Thus, for the viability of these tasks, efficient and accurate Skeletal Topography matching is needed. Pose matching is applied in Gait Recognition using techniques such as Silhouette correlation[26], Gait Energy Image methods[28,27,32] and model based matching approaches which involves modeling the pose as an inverted-pendulum model[29] or as stick-models[31].

3. System Design and Functions

In this application, we aim to develop an interactive application for teaching dance/ exercise moves to children and/or persons with learning disabilities. The application will be presented as a game where the user will be required to select a song or exercise routine from a predefined list. Once the user has made a selection they are asked to form a starting pose. Once the starting pose is formed, the lesson begins, albeit at a very slow pace so that new users do not get overwhelmed. In the lessons, the dance poses are shown to the use and the user has to match the same pose. For a desired pose, the system uses a camera to capture the image of the user. Then with the means of various processes of Pose estimation, feature extraction and normalization, the pose data is matched using a weighted matching scheme. The subsections below describe the process in detail.

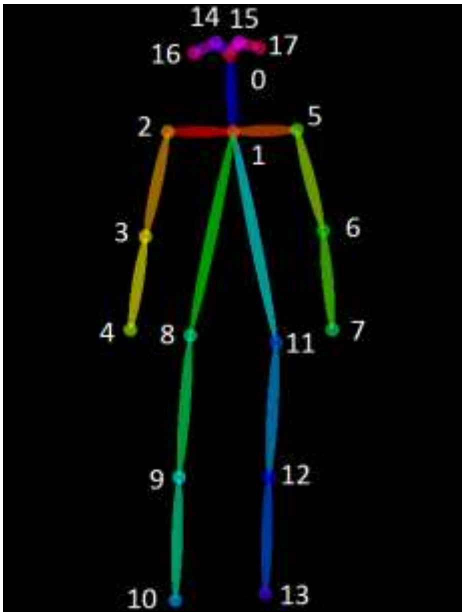

Fig. 2 - Pose Estimation Keypoints

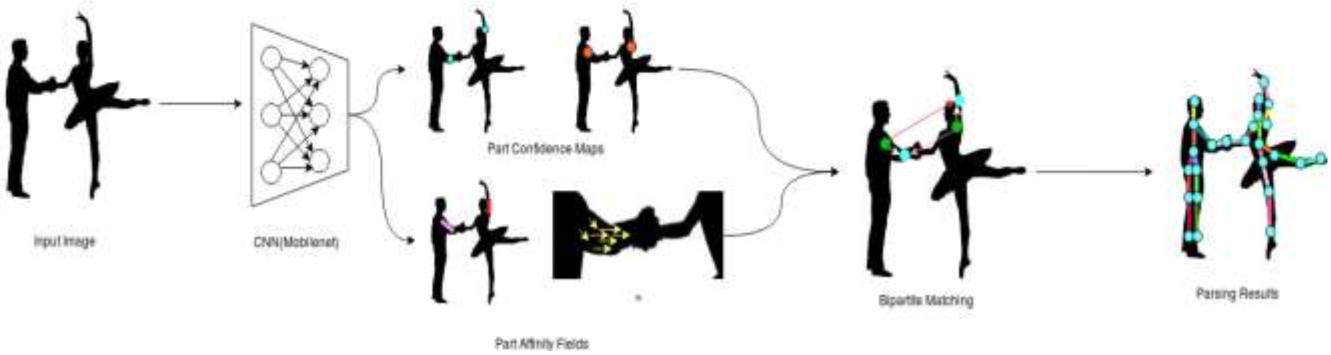

Fig-3 OpenPose Pipeline

3.1. *Pose Estimation through Deep Learning approach.*

In the first step of the process, we need to infer the positions of the key human body joints from a single 2D image. We use the OpenPose[3,4] deep learning library to jointly detect human body, hand and facial keypoints on single images. This library is based on pose estimation using Part Affinity Fields (PAFs)[5]. The system is real-time and its computational performance on body keypoint estimation is invariant to the number of detected people in the image. The code for the pose estimation model is written in Tensorflow[2].

In our work we use it to estimate 18 body keypoints corresponding to major human body joints. The processing pipeline makes use of a neural network to take in input image and gives out an output comprising of body part confidence heatmaps and Part affinity fields(PAFs). Using this data we compute the location of body parts and joints in the image. Bipartite graph matching and line integral using PAFs is done to ensure that all the joint components belong to the correct person among the multiple persons we are tracking and ensures their connectivity.

The inference result provides us with a data structure encapsulating the coordinates of respective body parts along with their confidence scores. These confidence scores will be later significant in matching the poses. The list of 18 keypoints are shown in Fig-2 and the keypoint indices are described in Table-1. We list only 14 out of the 18 keypoints as the remaining 4 keypoints, namely, Left-Ear, Right-Ear, Left-Eye and Right-Eye, describe the face features which are not required for our application. Furthermore, they may be a source of undesirable noise in the data causing inaccurate matching.

Table 1 - Keypoint descriptions of Pose Data.

Keypoint No.	Joint Name	Keypoint No.	Joint Name
0	Head/Nose	7	Left-Wrist
1	Neck	8	Right-Hip
2	Right-Shoulder	9	Right-Knee
3	Right-Elbow	10	Right-Ankle
4	Right-Wrist	11	Left-Hip
5	Left-Shoulder	12	Left-Knee
6	Left-Elbow	13	Left-Ankle

3.2. *Normalizing the estimated pose data.*

Differences in width and height of different persons can cause inconsistency in prediction of results. To ensure that our analysis structure is consistent and predictable, the resulting keypoints for each human can be represented as a vector in high dimensional space. We then use L2 normalisation to scale the vector to have unit magnitude. This ensures that all the keypoints during different detections are consistent with each other independent of width and height of the human

Fig-4 Estimated pose as seen by the application

After carrying out the normalization process we are left with vectors of pose data that are of unit norm. This will be useful when we compare pose data from wide varieties of body shape and sizes. The 28-Dimensional normalized pose vector can now be compared with a pre-recorded normalized reference vector to obtain the matchings.

3.3. *Pose Matching for scoring in lessons.*

The core application of pose estimation data is in the matching of poses to match the user pose to the reference correct pose. The resulting similarity metric can then be evaluated to generate the scores for the user. Previous approaches where the concepts of matching pose have been used, employed trigger based pose matching techniques[33]. These use additional hardware such as Proximity and Ultrasound sensors to acts as triggers when the correct configuration of pose is made according to the arrangement of sensors. This is approach is very limited in that it does not scale very well to the addition of new content. When new poses are added to the system, additional sensors would have to be added and the triggers reprogrammed. Thus, we need an intelligent approach that scales well to the addition of extra course material and can match any pair of poses.

As we know, the output of the pose estimation pipeline comprises of two types of data, the coordinates of the joints and their confidence scores of the detections. Matching solely on the basis of coordinates is not feasible as the coordinates of an accepted pose are not crisp, but rather a fuzzy search space. Thus, defining a fixed threshold does not work. Using vector matching metrics such as Cosine distance might work well as it is used in the fuzzy matching of strings[34]. We just need to represent the pose data in the form of a high dimensional vector. But, alone it is not sufficient as the process of pose estimation is subjective and different joints may be identified with varying degrees of confidence. As a result, joint predictions detected with very low confidence will get the same weightage as a joint prediction with high confidence. This will lead to noise in the calculation and mis-classifications. To counter this, we utilize the confidence scores from the model. We devise a weighted matching function where we assign confidence scores to the keypoint vector. In this way, the joints having higher confidence scores have greater contribution towards the distance metric compared to those with lower confidence scores. Therefore, for two normalized pose vectors P and Q, the weighted matching formula is defined as:

Fig-5 High level system overview

$$D(P,Q) = \frac{1}{\sum_{k=1}^{14} CS_k} \times \sum_{k=1}^{14} CS_k \left|\left|P_k - Q_k\right|\right| \tag{1}$$

In this formula, CS_k is the confidence score of the k_{th}keypoint of P. $P_k$and $Q_k$represent the x and y positions of the k_{th} keypoint of each vector.

The pose vector is matched to a gallery of pre-recorded poses which are stored in a database as normalized vectors. These pre-recorded vectors are labeled to identify them to which song and move they belong to. For a single pose made by the player it is continuously matched with the pre-recorded vectors and the label of the most matched vector is compared. If the label is correct, then the user is awarded points based on the degree of matching, otherwise they lose points.

4. Discussions

In order to evaluate various test scenarios**,** we first present the implementation level details of the software and how the various components of the software interact with each other. After clearing the basics of the software control flow we then present comparative study between the pose estimation capabilities offered by the pose estimation model in our application in contrast to that offered by a standard hardware solution such as Microsoft Kinect.

4.1. System Setup

As shown in **Fig-5**, the test setup for the system comprises of a web camera mounted on the display. The user is shown a real time video feed on the display on which the skeleton of the user is shown superimposed on the feed as seen in **Fig-4**. On the same display, adjacent to the video feed is the Task window which presents the dance move the player has to imitate. As the player tries to imitate the pose of the dance move, the skeletal data from the player is matched with the pose of the dance move in the Task window. The similarity score thus generated are mapped to a percentage between 0-100, where 0 means absolute no match while, 100 means perfect match. The percentage scores are then displayed to the user. The process of pose estimation for realtime skeleton stream and pose matching are performed continuously and the scores are displayed in real time.

Fig-6 (Clockwise from top) Kinect Skeleton stream; Kinect Depth Stream; OpenPose Skeleton stream; OpenPose output under partial occlusion.

4.2. Comparison with Kinect Skeleton Stream

A comparison of skeleton streams between Kinect and OpenPose based model in our system is shown in **Fig-6**. From the images it can be seen that the Kinect skeleton stream tracks 20 keypoints on the Skeleton while our system tracks 18 keypoints on the skeleton which also includes the 4 facial keypoints. The major advantage that our model presents is the fact that it can work under occlusion. Sideways profiles are very hard for kinect to produce skeleton on and in such cases the skeleton parts overlap in undesirable manner. Since the OpenPose model, on which our pose estimation pipeline is based, is rained on variety of images of daily objects and persons. It can infer the global context of each person's position. So, if a person is under partial occlusion, the model can accurately guess where the hidden part would most probably be based on the data that it has seen and can output the best guess on the position of keypoints on the occluded part.

The reason that the deep learning based approach is superior can be made apparent through a look into each of the method's internal workings. The brain of the Kinect which outputs the skeleton stream also uses a machine learning based approach called Random Decision Forests(RDF)[35,7,8] to generate part proposals. It takes input as the depth map and the RGB image and classifies each pixel in the combined RGB-D image based on the 20 keypoints corresponding to each part. This gives output as approximate locations of each keypoint part in the image. Then a mean shifting algorithm is performed to localize to the exact keypoint positions[7]. But, studies have shown[36] that the reliability of information in decision trees depends on feeding the precise internal and external data during training. The decisions contained in the tree are based on pre-assumed expectations and some irrational expectations can lead to flaws and errors in the decision process. Also, the learning capacity of such models is very low compared to a neural network. Therefore, in order to ingest the most information, a forest of such trees have to be trained which increases the computational complexity.

Table-2 Training Times for different methods.

Method	Training Hardware	Training Dataset	Training Time
Kinect(RDF)	1000 core cluster[7]	1 M RGB-D images[7]	24000 CPU hours or 1 day[7]
tf-pose-estimation	8 Tesla K80 + 48 CPU cores[4]	MS-COCO Keypoints-2016	1 day (approx.)[4]
Our(based on tf-pose-estimation)	1 Tesla P100 + 8 CPU cores	MS-COCO Keypoints-2017	80 hours or 4 days (approx.)

The deep learning model on the other hand makes use of complex yet efficient neural network architecture with many layers. It uses Convolutional Neural Networks[38] to extract useful features from the images. The features extracted are then used by another network on top of it to produce pose estimation results. In order to make the process fast and effective, many optimizations such as efficient backpropagation[40], residual connections between network layers[39], batch normalization[37] and artous convolutions[41] are used. Deep learning based methods have been shown to easily outperform other methods in almost all fields of computer vision and image understanding[42]. As suggested by [42] and [45], deep neural networks have lager and complex learning capabilities and require lesser number of parameters. Their stacked hierarchical layers can model the structure of the mammalian visual cortex upto a great extent[43]. Lower layers learn the most basic features such as lines and strokes, whereas, the higher layers use the information from the lower layer to model complex features such as curves, patterns, and facial structures[44].

The effectiveness and efficiency of deep learning models with respect to training and amount of training data required is shown in **Table-2.** Training the RDF model used in Kinect takes a full day on a distributed 1000 core cluster while the deep learning model used in[4] takes roughly the same time and requires a small cluster with total 48 cores and 8 Nvidia Tesla K80 GPUs. Even with our limited resources of 8 cores and 1 Nvidia Tesla P100, we were able to train the full model in about 80 hours on a Google Cloud Compute Instance at a cost of $1.33 per hour of usage. This shows that the cloud computing infrastructure is developed to a point where large compute power is available to general public at a low cost for such applications. The availability of better and open source algorithms, better infrastructure, more data makes deep learning based approaches have the potential to replace specialized hardware based solutions in vision based tasks.

5. Conclusion

We have developed a Human-Computer Interaction platform which jointly addresses the problems of physical growth of children and rote learning using an unified model based approach. Our method provides a gamified approach through which children can watch and learn new skills through body gesture based interaction. We have demonstrated the effectiveness of the proposed method on the software level. This study investigates the applications of pose estimation without using Kinect or any other specialized hardware and proposes a possible way to use deep learning method to extract 2D skeleton model from any image or real time video, which can be used in treating Learning disabilities and exercise training. Using such methods makes our system an inexpensive application of skeletal tracking technology which was only possible earlier with the use of Kinect. Thus, replacing the Kinect hardware and proprietary software drivers with a open source software model for pose estimation requiring only a single RGB camera. This approach opens up a wide field wherein many applications utilizing the features of proprietary technologies as used in Kinect can now be redesigned to make use of the state of the art human pose estimation libraries to perform the exact same tasks. But, now these applications would be cost-effective and accessible to a larger audience who do not have the means and methods to acquire and setup some specialized hardware.

In the near future, deep learning is going to revolutionize many fields of image processing and machine vision and will not be only limited to their domain but expand forth. We hope to see many such deep learning based software alternatives emerge to compete with the similar performing hardware.

REFERENCES

[1] Ian Janssen and Allana G LeBlanc. 2010. Systematic review of the health benefits of physical activity and fitness in school-aged children and youth. International journal of behavioral nutrition and physical activity 7, 1 (2010), 40.

[2] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., et al.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015) Software available from tensorflow.org.

[3] Tomas Simon Shih-En Wei Hanbyul Joo Gines Hidalgo, Zhe Cao and Yaser Sheikh. Open Pose. https://github.com/CMU-Perceptual-Computing-Lab/openpose

[4] Ildoo Kim. tf-pose-estimation. https://github.com/ildoonet/tf-pose-estimation

[5] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2017. Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017), 1302–1310.

[6] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. 2016. Convolutional Pose Machines. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016), 4724–4732.

[7] Shotton, Jamie and Fitzgibbon, Andrew and Blake, Andrew and Kipman, Alex and Finocchio, Mark and Moore, Bob and Sharp, Toby. 2011. Real-Time Human Pose Recognition in Parts from a Single Depth Image. IEEE Conference on Computer Vision and Pattern Recognition(CVPR) (2011)

[8] Ross Girshick, Jamie Shotton, Pushmeet Kohli, Antonio Criminisi, and Andrew Fitzgibbon. 2011. Efficient regression of general-activity human poses from depth images. In Computer Vision (ICCV), 2011 IEEE International Conference on Computer Vision. IEEE, 415–422.

[9] J. Han, L. Shao, D. Xu, and J. Shotton. 2013. Enhanced Computer Vision With Microsoft Kinect Sensor: A Review. IEEE Transactions on Cybernetics 43, 5 (Oct 2013), 1318–1334. https://doi.org/10.1109/TCYB.2013.2265378

[10] Min Sun and Silvio Savarese. 2011. Articulated part-based model for joint object detection and pose estimation. In Proceedings of the IEEE International Conference on Computer Vision. 723–730.

[11] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, and Kevin Murphy. 2017. Towards Accurate Multi-person Pose Estimation in the Wild. https://arxiv.org/abs/1701.01779

[12] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. 2014. Joint training of a convolutional network and a graphical model for human pose estimation. In Advances in neural information processing systems. 1799–1807.

[13] A. Bulat and G. Tzimiropoulos. Human pose estimation via convolutional part heatmap regression. In ECCV, 2016.

[14] X. Chen and A. Yuille. Articulated pose estimation by a graphical model with image dependent pairwise relations. In NIPS, 2014.

[15] V. Belagiannis and A. Zisserman. Recurrent human pose estimation. In 12th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG), 2017.

[16] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In ECCV, 2016.

[17] T. Pfister, J. Charles, and A. Zisserman. Flowing convnets for human pose estimation in videos. In ICCV, 2015.

[18] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. Efficient object localization using convolutional networks. In CVPR, 2015.

[19] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In CVPR, 2014.

[20] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V. Gehler, and Bernt Schiele. 2016. Deep-Cut: Joint Subset Partition and Labeling for Multi Person Pose Estimation. 2016 IEEE Conference on Computer Vision and Pattern Recognition(CVPR) (2016), 4929–4937.

[21] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele. Deepercut: A deeper, stronger, and faster multiperson pose estimation model. In ECCV, 2016.

[22] G. Papandreou, T. Zhu, L. Chen, S. Gidaris, J. Tompson, andK. Murphy. Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. CoRR, abs/1803.08225, 2018

[23] Xu, Ran, Agarwal, Priyanshu Kumar, Suren Krovi, Venkat N. Corso, Jason J., Combining Skeletal Pose with Local Motion for Human Activity Recognition. 2012. In Articulated Motion and Deformable Objects, Springer Berlin Heidelberg. p114-123.

[24] Martín Félez, Raúl & Alberto Mollineda, Ramón & Sánchez, José. (2011). Human Recognition Based on Gait Poses. 6669. 347-354. 10.1007/978-3-642-21257-4_43.

[25] Matovski, Darko S., Mark S. Nixon, and John N. Carter. "Gait Recognition." Computer Vision. Springer US, pp 309-318, 2014.

[26] Phillips, P.J.; Sarkar, S.; Robledo, I.; Grother, P.; Bowyer, K., "The gait identification challenge problem: data sets and baseline algorithm," in Pattern Recognition, 2002. Proceedings. 16th International Conference on, pp. 385-388, 2002.

[27] Bashir, K.; Tao Xiang; Shaogang Gong, "Gait recognition using Gait Entropy Image," in Crime Detection and Prevention (ICDP 2009), 3rd International Conference on, pp.1-6, 3-3 Dec.2009.

[28] Jinguang Han and Bir Bhanu. 2006. Individual recognition using gait energy image. IEEE Transactions on Pattern Analysis & Machine Intelligence 2 (2006), 316–322.

[29] Taku Komura, Akinori Nagano, Howard Leung, and Yoshihisa Shinagawa. 2005. Simulating pathological gait using the enhanced linear inverted pendulum model. IEEE Transactions on Biomedical Engineering 52, 9 (2005), 1502–1513.

[30] James Little and Jeffrey Boyd. 1998. Recognizing people by their gait: the shape of motion. Videre: Journal of computer vision research 1, 2 (1998), 1–32.

[31] Yoo, J, H; Nixon, M, S; Harris, C, J, "Extracting gait signatures based on anatomical knowledge," Proceedings of BMVA Symposium on Advancing Biometric Technologies, pp. 596-606, 2002.

[32] C., Wang; J., Zhang; J., Pu; X., Yuan; and L., Wang, "Chrono-Gait Image: A Novel Temporal Template for Gait Recognition," Proc. European Conf. Computer Vision, pp. 257-270, 2010.

[33] S. Nandi, S. Deb, and M. Sinha. 2016. Augmented Exer-Learning Tool Using Ultrasonic Depth Visualization of Movement. In 2016 IEEE Eighth International Conference on Technology for Education (T4E). 242–243. https://doi.org/10.1109/T4E.2016.060

[34] Aminul Islam and Diana Inkpen. 2008. Semantic text similarity using corpus-based word similarity and string similarity. ACM Transactions on Knowledge Discovery from Data (TKDD) 2, 2 (2008), 10.

[35] Tin Kam Ho, "Random decision forests," Proceedings of 3rd International Conference on Document Analysis and Recognition, Montreal, Quebec, Canada, 1995, pp. 278-282 vol.1. doi: 10.1109/ICDAR.1995.598994

[36] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," in IEEE Transactions on Systems, Man, and Cybernetics, vol. 21, no. 3, pp. 660-674, May-June 1991. doi: 10.1109/21.97458

[37] Ioffe, Sergey and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.” ICML (2015).

[38] Krizhevsky, Alex, Ilya Sutskever and Geoffrey E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks.” NIPS (2012).

[39] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

[40] LeCun, Yann A., Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. "Efficient backprop." In Neural networks: Tricks of the trade, pp. 9-48. Springer, Berlin, Heidelberg, 2012.

[41] Howard, Andrew G., Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." arXiv preprint arXiv:1704.04861 (2017).

[42] Henry W. Lin and Max Tegmark, Why does deep and cheap learning work so well?, arXiv:1608.08225

[43] HUBEL, D. H., & WIESEL, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. The Journal of physiology, 160(1), 106-54.

[44] Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." In European conference on computer vision, pp. 818-833. Springer, Cham, 2014.

[45] The Unreasonable Effectiveness of Deep Learning. Yann LeCunn. https://www.cs.tau.ac.il/~wolf/deeplearningmeeting/pdfs/lecun-20141105-tau-intel-master-class.pdf