DEEP LEARNING BASED METHOD FOR 3D HUMAN POSE ESTIMATION FROM 2D FISHEYE IMAGES

[1] *Ching-Chun Chen* (*陳靖淳*), [2] *Chia-Min Wu* (*吳佳珉*) , [3] *Bing-Yu Chen* (*陳炳宇*)

[1,2,3] Communications and Multimedia Laboratory, Graduate Institute of Networking and Multimedia, National Taiwan University, Taiwan
E-mail: r04944003@ntu.edu.tw

ABSTRACT

In this study, we propose a deep learning based method to directly predict the human joint positions in 3D space from 2D fisheye images captured in an egocentric manner. The core of our method is a modified Inception-v3 convolutional neural network featured the larger convolution filter size, parameter reduction with SELU activation function, long short-term memory module, and the anthropomorphic constraints on the training loss. We also conduct four experiments to study the different effects upon the validation results when using different training settings of our work. The experience of our study may be helpful to develop more complicated deep learning network in a reasonable resource requirement to deal with the computer vision problems.

Keywords *Fisheye Image; 3D Human Pose Estimation; Egocentric View; Convolutional Neural Networks; Inception; LSTM; SELU; Anthropomorphic Constraints;*

1. INTRODUCTION

In recent years, there have been some remarkable breakthroughs in the research fields like the computer vision and the virtual reality. Although some companies have launched their first virtual reality HMDs (Head-Mounted Display), the lack of high accuracy full-body pose tracking techniques makes the adoptable human-machine interaction mechanisms very limited. To deal with such problem, exploiting the strong power of the deep learning in the computer vision field may be the most possible direction while trying to find the best solution. In this study, we attempt to propose a deep learning based method to directly predict the human body joint positions in 3D space from the 2D fisheye images captured in an egocentric manner.

2. RELATED WORK

We used the fisheye image datasets published by Rhodin et al [1] to conduct our research experiment. They proposed a framework that combined the deep learning based prediction of human joint 2D coordinates with the ray-casting based post-processing algorithm to deal with the application used to track the human 3D poses from the 2D fisheye video captured in an egocentric manner. Chunyu Wang et al [2] also published their work on how to combine deep learning with traditional computer vision algorithms to reconstruct the human poses in 3D space however they targeted on normal images captured by the ordinary perspective camera. To be the successor of Googlenet [3], Szegedy et al [4] proposed their refined architecture named Inception-v3 with several design principles provided by them. Besides the powerful LSTM module invented by Hochreiter et al [5] to deal with the gradient vanishing problem in recurrent neural networks, SELU activation function was also a distinguished deep learning tool proposed by Klambauer et al [6] to make a self-normalizing neural network.

3. DEEP LEARNING BASED METHOD

Our deep learning based method is designed to be able to estimate human 3D poses from only 2D fisheye images. Generally speaking, our target is to make a very simple workflow that only involves 2D images as input, human body joint positions in 3D space as output, and the deep learning network as the only processing unit, without any other non deep-learning post-processing algorithms.

To handle such a task which heavily depend on visual features of full color bitmap images with large size, convolutional neural networks are the best choices for us. Since Alexnet [7] proved to the world that convolutional neural network with 8 layers could be not only efficient but also accurate on image category classification with the computational power of modern graphics processing units, there have been more and more outstanding CNN networks proposed by those top research teams.

Among these CNN networks with different architecture and characteristics, we have found that Inception-v3 [4] proposed by Google as the successor of

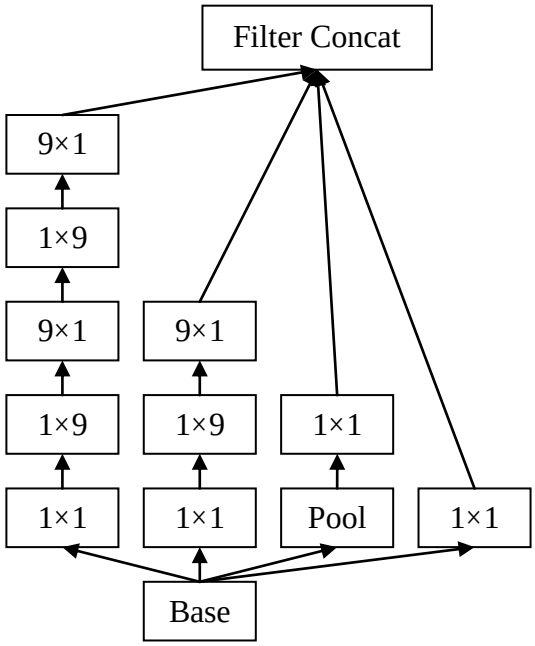

Figure 1: We set $n = 9$ instead of the default $n = 7$ in the Inception module proposed as Figure 6 in [4].

GoogleNet is very suitable to process 2D fisheye images, the reason will be discussed in the paragraph under subheading 3.1.

To sum up, the convolutional neural network which we use in the workflow to predict 3D human pose from 2D fisheye images is mainly based on the Inception-v3. However, instead of keeping the original architecture be intact, we not only design the modified architecture for Inception-v3 but also use some performance enhanced techniques to make the network perform better on the job we want it to do.

All the changes we have done to the Inception-v3 can be categorized into four groups: convolution filter size, parameter reduction with SELU, long short-term memory, and anthropomorphic constraints. We will give explanation in detail under subheading 3.1, 3.2, 3.3, and 3.4.

Besides these main changes, we also remove the fully-connected layer for logits and the softmax layer for classification from the last part of Inception-v3 architecture because that these layers are designed to handle classification problems. We use the regression operations in the end of our modified Inception-v3 to finish the prediction of 17 human joint positions in 3D space. To get all of the 17 joint positions, our last regression operation also has 17 sets of weight and bias variables for the corresponding joints.

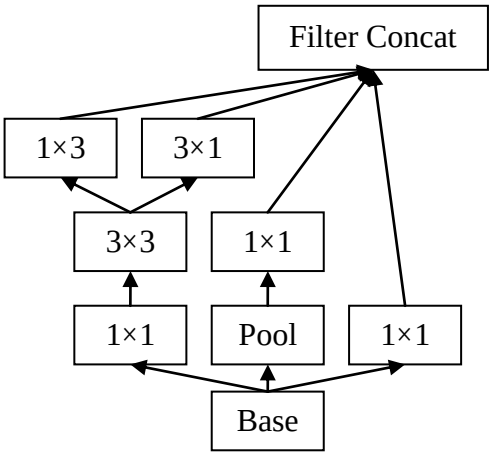

Figure 2: We remove the second branch from the Inception module proposed as Figure 7 in [4].

3.1. Convolution Filter Size

The original Inception-v3 architecture has some very important structures that will be very powerful to process 2D fisheye images. According to the optimization principles proposed by Szegedy et al [4], the larger filter size used in spatial aggregation of convolutional layers can be factorize to smaller filter size without expecting serious adverse effects because there will not be much loss in representational power if we can perform the factorizing operations correctly.

For example, performing the 5×5 convolution operation once is equivalent to performing the 3×3 convolution operation twice. This is due to the first 3×3 convolution operation with stride length 1 covering the same processing area of the 5×5 convolution operation can just form a suitable processing area for the second 3×3 convolution operation and finally generate a 1×1 spatial aggregation having the similar representational power to the spatial aggregation generated by just doing the 5×5 convolution operation once.

On the basis of such principle, even one convolution operation with the $n \times n$ filter size can be transformed into two parallel or sequential convolution operations with $1 \times n$ filter size and $n \times 1$ filter size respectively. Although this kind of asymmetric filter shape is originally designed to target the lower parameter count and the lower computational power requirement, we think prediction of human joint positions in 3D space from 2D fisheye images can also benefit from it. Unlike normal images captured by standard perspective cameras, straight lines in the real world will become curves in fisheye images when they are captured by fisheye cameras. As a result, visual features in fisheye images are much more difficult to be learned by symmetric convolution filter shapes with disadvantages in curvature analysis abilities. Asymmetric convolution

filter shapes in the other hand can extract visual features from curves in fisheye images more easily.

Finally, based on the Inception module proposed as Figure 6 in [4], we decide to set n = 9 instead of the default n = 7 for such structure (see Figure 1). By setting the longer convolution filter size, the larger area (17 × 17 for the first branch and 9 × 9 for the second branch in Figure 1) can be processed when performing convolutional operations in our neural network.

3.2. Parameter Reduction with SELU

Under this subheading we first explain what SELU is and then we will show how SELU plays an important role in our modified Inception-v3. Proposed by Klambauer et al [6], "SELU" is the abbreviation of "Scaled Exponential Linear Units". The most striking feature of SELU is that this is the first activation function which can let a neural network be self-normalizing.

Why is it important for a deep learning network to perform normalization operation on data samples even between layers and layers? Neural networks trained on normalized data samples will have stabilizing gradients between training iterations. Therefore, such neural networks can reach the state of convergence much faster than neural networks trained on data samples without normalization. Besides, the annoying gradient vanishing problems or gradient explosion problems can also be mitigated by letting data samples be normalized.

There have been other methods proposed by some research teams trying to achieve the same target. The batch normalization network [8] is one of the best in networks which make data samples normalized between layers and layers. Generally speaking, the batch normalization network has a whole layer designed especially for data samples normalization between layers and layers. Although the batch normalization technique can perform very well on convolutional neural networks and recurrent neural networks, success stories of standard feed-forward neural networks with the batch normalization technique are rare. Lacking the weight sharing feature which is present at convolutional neural networks and recurrent neural networks, the standard feed-forward neural networks suffer from perturbations such as stochastic gradient descent (SGD) and stochastic regularization (Dropout).

After adding the LSTM module that we will give explanation in detail under subheading 3.3, the parameter count and the memory requirement of Inception-v3 without architecture modification increase a lot. To handle such a situation, we decide to remove the second branch from the Inception module proposed as Figure 7 in [4] (see Figure 2). However, while this kind of method can reduce the parameter count and the memory requirement to the same amount before adding LSTM, the prediction accuracy on the validation set will also decline.

Finally, we have found that replacing the ReLU activation functions and the batch normalization techniques used in our modified Inception-v3 with just the SELU activation functions can strike a balance between resource requirement (the parameter count and the memory requirement) and prediction accuracy.

Setting the same fixed point in [6] that the mean value of data samples at 0 and the variance value of data samples at 1, we have the formula below to perform the SELU operation with the 2 float point parameters calculated by [6].

$$\text{selu}(x) = 1.0507 \begin{cases} x & \text{if } x > 0 \\ 1.6733e^{x} - 1.6733 & \text{if } x \leq 0 \end{cases} \quad (1)$$

To achieve the initialization requirement of SELU, we also change the initial STDDEV value of all the weight variables to sqrt(1/n) in our modified Inception-v3.

3.3. Long Short-Term Memory

Known as LSTM, this technique has been widely used since it was proposed in 1997 by Hochreiter et al [5]. Although LSTM is mainly based on the idea of recurrent neural networks, its structure with multi-gates and cell state focuses on solving the gradient vanishing problem and the gradient explosion problem between the hidden layers at different timestamps.

The cell state C(t) in LSTM is used to save status information at timestamp t. The forget gate in LSTM is used to decide how much information recorded by the C(t-1) should be retained and inherited by the C(t). The input gate in LSTM is used to decide what kind of new information should be added to the C(t) and this gate also determines the strength of new information. Finally the output gate in LSTM is used to generate the hidden layer output at timestamp t.

Images which are used to train or predict the human poses in 3D space are usually captured as video frames. As a result, such an image dataset possesses many sequential images. LSTM and recurrent neural networks perform very well on predicting results from sequential input data samples. We have found that adding the LSTM module before the final regression operations of our modified Inception-v3 can improve the prediction accuracy on the validation set despite the increased memory requirement which can be mitigated by the method we propose under subheading 3.2.

3.4. Anthropomorphic Constraints

Inspired by the post-processing algorithm proposed by Chunyu Wang et al [2], we think the anthropomorphic constraints can be fused with our deep learning network. When predicting human joint positions in 3D space from 2D images, the torso joints and the head joints are usually easier to be predicted than other human body joints. The reason is that there is not much difference in

3D position of the torso joints or the head joints between different human poses especially when the images are captured in an egocentric manner. On the other hand, the 3D positions of the joints of human limbs always vary a lot with different human poses.

To fuse the anthropomorphic constraints with our deep learning network, we design a weight parameter for every human limb joint. This weight parameter will be used to determine how strong will the loss value (error distance between the predicted value and the ground truth value) of a single limb joint affect the final aggregated loss value from all the body joints while training the neural network:

$\text{Loss}_{Total} := 0$
For each body joint j in the limb set:
$$\text{Loss}_{Total} := \text{Loss}_{Total} + \text{Weight}_j \times \text{Loss}_j \quad (2)$$
For each body joint k not in the limb set:
$$\text{Loss}_{Total} := \text{Loss}_{Total} + \text{Loss}_k$$

Where the joints in the limb set are the joints with ticks in the following table (we use the same human body joints definition as EgoCap [1]):

Table 1: The joints in the limb set.

Joint Name	*Limbs*
Neck	
Left Shoulder	
Left Elbow	✓
Left Wrist	✓
Left Finger	✓
Right Shoulder	
Right Elbow	✓
Right Wrist	✓
Right Finger	✓
Left Hip	
Left Knee	✓
Left Ankle	✓
Left Toe	✓
Right Hip	
Right Knee	✓
Right Ankle	✓
Right Toe	✓

4. EXPERIMENT RESULT

All of the following experiment results are obtained with our modified Inception-v3 trained and validated on the sequence of 750 2D fisheye image sets of gesturing and interaction provided by EgoCap [1] with joint position ground truth in 3D space.

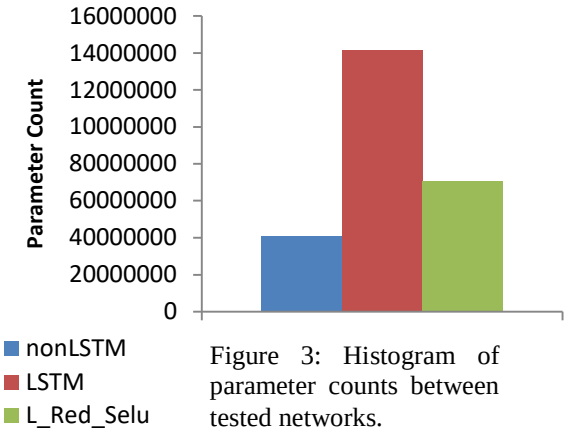

Figure 3: Histogram of parameter counts between tested networks.

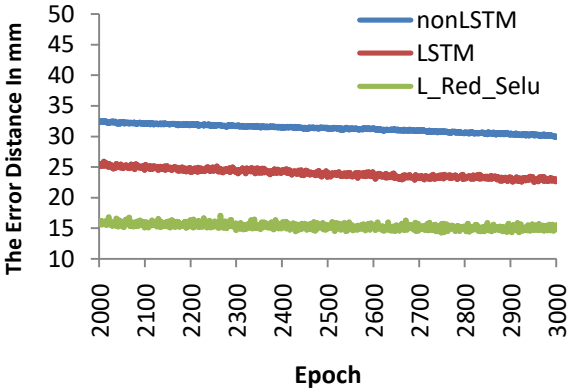

Figure 4: The curves of average error distance per joint between epoch 2000 and epoch 3000. "L_Red_Selu" means the parameter reduction version of Inception-v3 with LSTM and SELU.

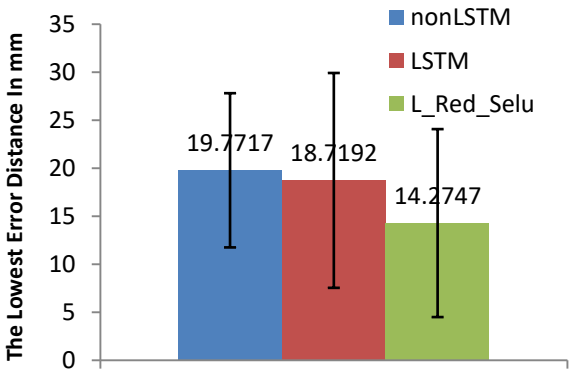

Figure 5: The lowest mean value of average error distance per joint which the tested networks can reach. The black error bars represent the standard deviation.

4.1. Experiment Environment

In the 750 2D fisheye image sets, one image in each image set is captured by the left fisheye camera setting on the head of the subject in an egocentric manner, the

other is captured by the right fisheye camera at the same time. To preprocess the dataset before we start the experiments, we first crop the left-eye images and the right-eye images to the same size of 1000 × 1000, and then resize them to the size of 331 × 331 that slightly larger than the modified Inception-v3 input size of 299 × 299 for random cropping when training. After the above-mentioned procedure, all of the images will be processed with the minmax_scale(-1, 1) function of scikit-learn (a python library) to finish the initial normalization. Finally, we divide the sample image sets to 750 / T groups, and then we randomly choose 60 / T groups of image sets to be the validation dataset. The reaming groups of image sets as the training dataset will be used to train our modified Inception-v3. The parameter T is prepared for our LSTM module when training and validating, we set it to 5. Tesla P100 with 16GB memory and Tesla K80 with 12GB memory (one core) are the GPUs we used to train and validate on the dataset with our modified Inception-v3. In the following experiments, we define the mean value of average error (Euler distance in millimeter between the ground truth and the prediction) per joint across all the validation samples as the validation accuracy in one training epoch. In the end, we compare the performance of different training settings with each other according to the highest validation accuracy they can reach.

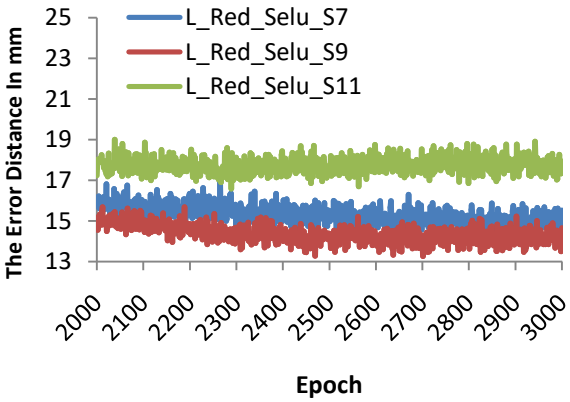

Figure 6: Curves of average error distance per joint between epoch 2000 and epoch 3000. “S9” means kernel shape *n* = 9.

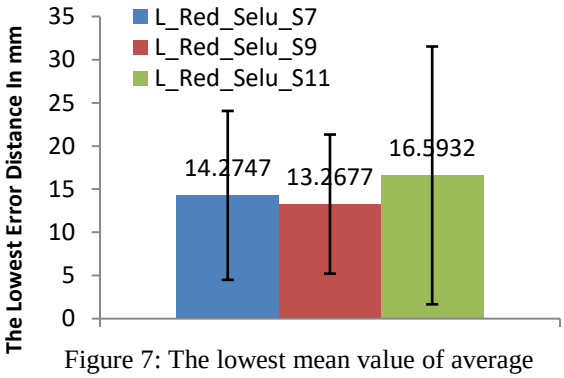

Figure 7: The lowest mean value of average error distance per joint.

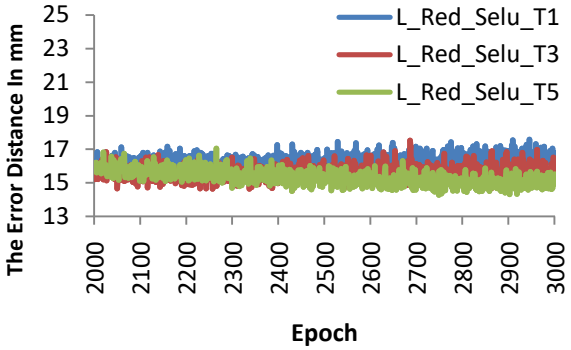

Figure 8: Curves of average error distance per joint between epoch 2000 and epoch 3000. “T5” means period parameter T = 5.

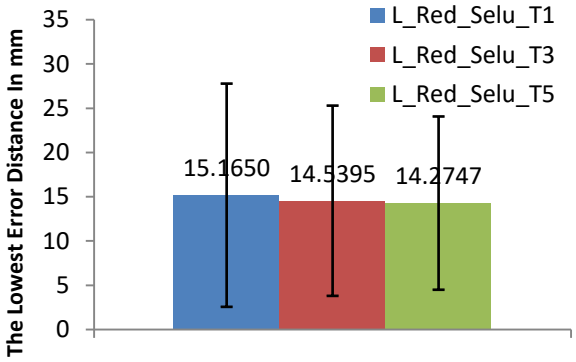

Figure 9: The lowest mean value of average error distance per joint which the tested networks with different T can reach.

4.2. Evaluation

In the first experiment, we compare the performance between the original Inception-v3 without the LSTM module, the original Inception-v3 with the LSTM module (set the period parameter T as 5), and the parameter reduction version of Inception-v3 with LSTM and SELU (set T as 5). All the other training parameters and experiment variables of them remain the same. We set the batch size as 30, the learning rate as 1e-5, the max training epoch as 10000 (3000 for the reduction version with SELU and LSTM), the convolution filter size as default, and the anthropomorphic constraint weight for limb joints as 1.4. Although the parameter count of the original Inception-v3 with the LSTM module is 3.478 times (see Figure 3) as great as the parameter count of the original Inception-v3 without the LSTM module, the former is not only trained much faster (see Figure 4) but also more accurate than the latter by 5.62% (see Figure 5). Among the three networks validated, the parameter reduction version of Inception-v3 with LSTM and SELU has the modest parameter count (see Figure 3) and the fastest training speed (see Figure 4) while being the most accurate.

(More accurate than nonLSTM by 38.5%; More accurate than LSTM by 31.1%. See Figure 5)

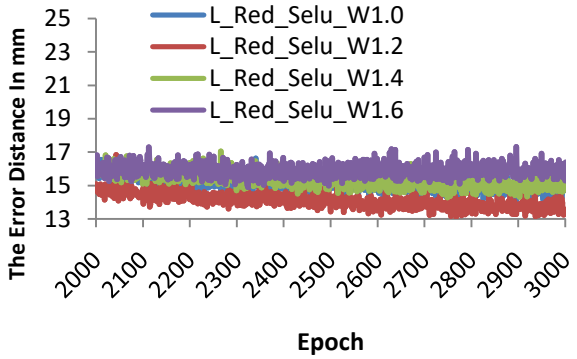

Figure 10: Curves of average error distance per joint between epoch 2000 and epoch 3000. "W1.2" means constraint weight = 12.

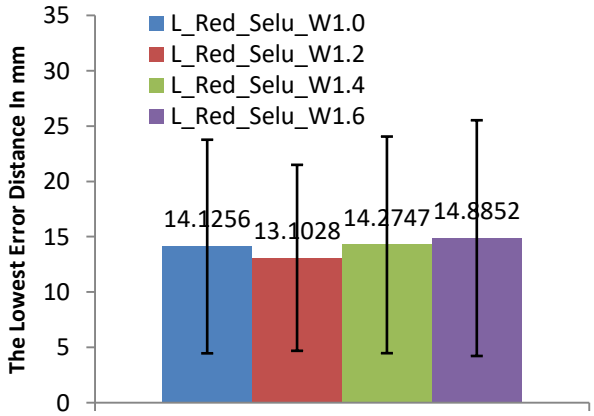

Figure 11: The lowest mean value of average error distance per joint which the tested networks with different weights can reach.

In the second experiment, we compare the performance between the parameter reduction version of Inception-v3 networks with LSTM, SELU, and different convolution filter shapes (7, 9, and 11). All the other training parameters and experiment variables of them remain the same. We set the batch size as 30, the learning rate as 1e-5, the max training epoch as 3000, the period parameter T as 5, and the anthropomorphic constraint weight for limb joints as 1.4. Among the three networks validated, the parameter reduction version of Inception-v3 network with LSTM, SELU, and the convolution filter shape of 9 has the fastest training speed (see Figure 6) while being the most accurate. (More accurate than the shape of 7 by 7.6%; More accurate than the shape of 11 by 16.2%. See Figure 7)

In the third experiment, we compare the performance between the parameter reduction version of Inception-v3 networks with LSTM, SELU, and different period parameters T (1, 3, and 5). All the other training parameters and experiment variables of them remain the same. We set the batch size as 30, the learning rate as 1e-5, the max training epoch as 3000, the convolution filter size as default, and the anthropomorphic constraint weight for limb joints as 1.4. Among the three networks validated, the parameter reduction version of Inception-v3 network with LSTM, SELU, and the period parameter T of 5 has the fastest training speed (see Figure 8) while being the most accurate. (More accurate than the T of 3 by 1.9%; More accurate than the T of 1 by 6.2%. See Figure 9)

In the fourth experiment, we compare the performance between the parameter reduction version of Inception-v3 networks with LSTM, SELU, and different anthropomorphic constraint weights for limb joints (1.0, 1.2, 1.4, and 1.6). All the other training parameters and experiment variables of them remain the same. We set the batch size as 30, the learning rate as 1e-5, the max training epoch as 3000, the convolution filter size as default, and the period parameter T as 5. Among the four networks validated, the parameter reduction version of Inception-v3 network with LSTM, SELU, and the constraint weight of 1.2 has the fastest training speed (see Figure 10) while being the most accurate. (More accurate than the weight of 1.0 by 7.8%; More accurate than the weight of 1.4 by 8.9%; More accurate than the weight of 1.6 by 13.6%. See Figure 11)

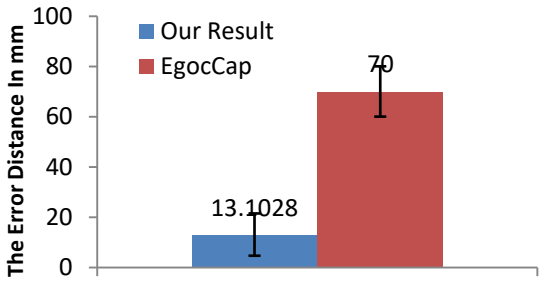

Figure 12: Comparison between our prediction error distance and the prediction error distance of EgoCap on the same dataset.

Compared with the accuracy reached by EgoCap [1] on predicting human joint 3D positions from the same sequence of 750 2D fisheye image sets, our work is more accurate by 434.2%. (see Figure 12)

Finally, we draw the prediction of the joint positions and the ground truth of the joint positions in the same time to make our readers know what accuracy our work can reach at a glance. (See Figure 13)

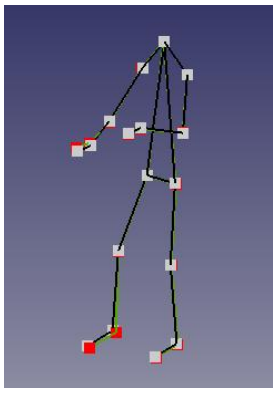

Figure 13: The gray points and the black lines are the ground truth. The red points and the green lines are our predictions. The points represent the joints while the lines represent the bones between the joints.

5. CONCLUSIONS

To find the deep learning based method that can directly predict the human body joint positions in 3D space from the 2D fisheye images captured in an egocentric manner, we have proposed the modified architecture of the Inception-v3 convolutional neural network with some performance enhanced techniques such as LSTM, SELU, and anthropomorphic constraints.

Featured with the modest parameter count and the memory requirement, our work has shown that it can reach higher accuracy than the original Inception-v3 in our first experiment. In other experiments, we have also demonstrated the different effects upon the validation results when using different training settings of our work. Compared with the EgoCap [1], our work can be much more accurate when predicting on the same fisheye image dataset.

REFERENCES

[1] H. Rhodin, C. Richardt, D Casas, E Insafutdinov, M Shafiei, H. P. Seidel, B. Schiele, and C. Theobalt, "EgoCap: Egocentric Marker-less Motion Capture with Two Fisheye Cameras", *ACM Transactions on Graphics (TOG)*, Vol. 35, No.162, 2016.

[2] C. Wang, Y. Wang, Z. Lin, A. L. Yuille, and W. Gao, "Robust Estimation of 3D Human Poses from a Single Image", In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1106–1113, 2014.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions", In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[4] C. Szegedy, V. vanhoucke, S. Ioffe, and J. Shlens, "Rethinking the Inception Architecture for Computer Vision", In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[5] S. Hochreiter, and J. Schmidhuber, "Long Short-Term Memory", *Neural Computation*, Vol. 9, pp. 1735-1780, 1997.

[6] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-Normalizing Neural Networks", In *Neural Information Processing Systems (NIPS)*, 2017. (Accepted)

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", In *Neural Information Processing Systems (NIPS)*, pp. 1097-1105, 2012.

[8] S. Ioffe, and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift", In *International Conference on Machine Learning (ICML)*, pp. 448-456, 2015.