Augmented Human Action Classification with Joint Estimation via double CNN

Colin Zheng
Mechanical Engineering
Stanford University
colinz07@stanford.edu

Yixian Li
Mechanical Engineering
Stanford University
s031830@stanford.edu

Tianheng Shi
Mechanical Engineering
Stanford University
ts5@stanford.edu

Abstract

In this project, we proposed an augmented human action classification model with human joint keypoint information. To approach this final goal of our project, we completed two intermediate goals: the first goal was to implement a single-person joint localization model, in which we explored three models: the CDRP model which we implemented in our own, the ResNET 34 model similar to DeepPose [8] and a stacked hourglass model as benchmark. The validation accuracy using mean PCKh@0.5 metric is around 26 for baseline and 34 for ResNET34, while the benchmark model reaches around 82. The second goal was to incorporate the keypoint information to the classifier model. For the joints embedding layer, we implemented both linear and convnet methods. The baseline model for the classifier was a CNN model with only the image as input, which we included for comparison purposes. The validation accuracy for baseline model was 48.34%, while the models with linear and convnet joints embedding layer had validation accuracies of 52.19% and 63.53% respectively. We concluded that including joints information was beneficial to the model performance, and that the convnet embedding layer outperformed the linear as it included information of joint connections. Lastly, the limitations of our model and the dataset were identified, and possible future work was discussed.

1. Introduction

Image classification task is one of the most important and widely implemented tasks in computer vision. Its implementation varies from classifying cat images to detecting vehicles for autonomous driving cars. Within the various applications in image classification, the topic of human action classification remains a challenging task [1]. The target of human action recognition is to classify different actions from a sequence of observations and different environmental conditions. In this project, an image database named MPII [2] is used for such classification task for single person. The applications for action recognition models include surveillance and assisted living, healthcare monitoring, human-robot interaction and so on. Hence, an accurate model in human action recognition has a potentially significant impact on these various downstream models that would improve various aspects of the human life.

Traditional effort have been taken to utilize CNN or RNN networks for action classification purposes. Other methods to improve the model include generating shape information and subtracting backgound of the image [1]. However, for action recognition, we think that accurate interpretation of the human pose is also important as it is directly linked to the action. Our team propose that the action classification model can be augmented with the inclusion of human joint keypoint information. Our classification model first estimates the joint keypoint locations, and the information is then added to the image features to recognize different actions. By performing this act, we hope we could get an improved performance over some traditional single CNN or RNN methods.

1.1. Problem Statement

The goal of the project is to produce an augmented CNN model that could estimate the joint location and classify the action of a single person given an input image. This goal consists of two steps: the joint localization step is to identify the pose keypoints from the input images through an upstream CNN model, and the action classification step is to classify the images into different pose classes, making use of the intermediate keypoints features in multiple ways through a downstream CNN model. An image representation of the workflow is shown in Fig. 1. The main difference is that the figure referred shows 3D keypoints, whereas our team used 2D keypoint estimations.

1.2. Related Work

There has been various work related to both human joint estimation and human pose classification. One of the earliest achievement on joint estimation via CNN is DeepPose from [8]. It used a 7-layer CNN model with a L2 regression loss on the key-point positions within a image con-

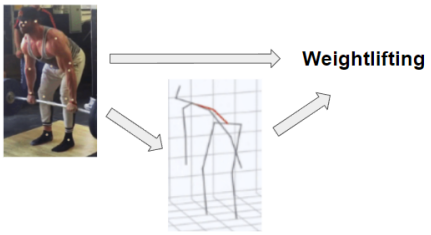

Figure 1. Workflow from project statement. Weightlifting image and keypoint source: https://www.kdnuggets.com/2020/08/3d-human-pose-estimation-experiments-analysis.html

taining a single person pose. Besides, it reused the joint location prediction from the the previous output by selecting a certain sub-region around these locations and repeated the training process for a series of stages. Aside from the regression-based approach, the later developed detection-based approach gives better performance in joint estimation. One of the most significant works is the introduction of DeepCut from [7]. It proposes a method to jointly detect the object's body parts while labelling the locations based on a pairwise probability scheme. The method is further improved by DeeperCut from [4], where a ResNET model [3] was utilized to give a better result on not only single-person but also multi-person joint estimation. Besides, the team also investigated into a method proposed by [6]. This model is called "Stacked Hourglass", the CNN architecture of which is designed like many hourglass stacked together where features are being maxpooled to a 1x1 convolution cell and being upsampled back repeatatively. With a L2 regression loss defined upon the heatmap representing a 2d gaussian space centered on location, the model achieved accuracy (on PCKh@0.5) above 0.85 consistently. This model serves as a benchmark to compare with our own model's performance.

Some of the most successful human pose classification methods include a SVM classifier by Yang et al. [11], which makes uses human poses as latent variables. This approach has similar underlying intuition as our project. Instead of using joints locations directly, human poses are seperated into four parts: upper body, left arm, right arm, and legs. The pose information was stored in a table called poselet, with correspoding action labels. Then the SVM scores are combined with the poselet information as the final scores. Such a SVM model could be tried to replace our currently implemented CNN/DNN classifier with softmax.

1.3. Dataset

The MPII dataset [2] is used for both the joint localization and activity classification tasks. The dataset includes both images and annotation for the images as a .mat file. The annotation file included keypoints labels, activity category classes, and information for other tasks such as bounding box coordinates, and video timestamp for video tasks. For the purpose of our project, only the keypoint locations and activity categories are used. The dataset has 25,000 images.

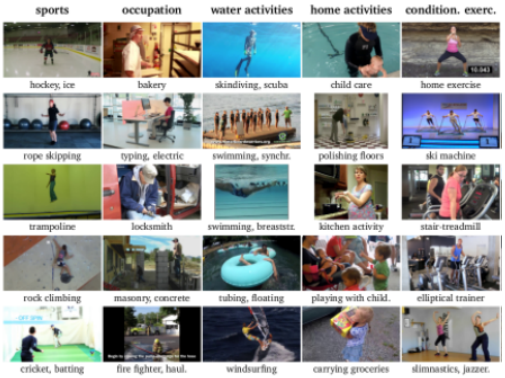

Figure 2. MPII Human Pose Dataset [2]

2. Technical Approach

We break down our technical approach into the data preprocessing, network design, metric, and experiment details.

2.1. Data Preprocessing

The image data from MPII dataset [2] has the same height and width dimension. In order for the team to build a adequate network that requires smaller image input feature with same shape across spatial dimensions, they are resized to 224 * 224. Since input features that represent intensities are always positive values, we zero-centered image by subtracting the mean value across the spatial dimensions within each image, each channel. We did not normalize the data since the range of intensity for each channel is the same. No data augmentation method was used.

For joint data that is both used to label the upstream localization model and train the downstream classifier model, we extracted three pieces information for each individual keypoint: the x pixel location, y pixel location, and a boolean visibility flag that indicates whether the keypoint is visible or not. The x and y pixel location was resized by the amount calculated from the image transformation above. Then they are both subtracted by 112 to make sure the range of the joint location is from -112 to 112, negating any downside from having all positive value in the input.

Moreover, after error analysis for the classifier model, it was noticed that for 5 classes of activity categories, the data after preprocessing was limited, which was less that 5% than the class that has the most data. Since data augmentation techniques like random cropping wouldn't work here due to the fact that joints locations must be specified, these

5 classes data was removed from the final experiments. The resulting dataset size was about 11,000.

80 percent of the data goes to training, while 10 percent is used for evaluation, another 10 for testing.

2.2. Network Design

2.2.1 Joint Localization Model

For joint detection, We first implemented a baseline model that utilized a self-built network (CDRP) and a regression loss. The network structure and loss function are similar to that proposed in [8]. Concretely, we used a network structure of Conv-Drop-ReLU-Pool2 four times until we spatially reduce the height and width to 14*14, and increase the channel size to 128. The Conv layers are designed with the padding of one and stride of one to maintain the spatial structure. We then flattened out the output and fed into four fully-connected layers (each with Batchnorm layer ahead). The output is a size 32 vector, representing 16 key-points' x and y positions. The l2 regression loss for a single training example x is computed as:

$$L = \frac{1}{m}\sum_{i=1}^{m}(y_i - f(w_i; x))^2 \quad (1)$$

where $m <= 32$ is the number of all labelled key-points, y_i is the predicted result for a certain joint location in one dimension, and w_i is the associated training weight.

The second model we have implemented utilized a pre-trained ResNET34 model and a regression loss. The ResNET model is then connected with a bottleneck fully connected layer that outputs a size 32 vector that represents the 16 keypoints' x and y locations. The same L2 loss was used. Both network structures are visually presented in Fig. 3 and Fig. 4.

The team also investigated into the 8-Stacked Hourglass network introduced in the literature review. It is claimed that a mean PCKh@0.5 (introduced in 2.3) accuracy of 0.8864 had been reached before. This method is only used as a benchmark to evaluate the performance of our own models. The team did not contribute any code for the benchmark model. The source code was referenced from [10].

2.2.2 Activity Classification Models

For the classifier model, two models were implemented. Two model structures are highly correlated. Their structures are visualized as Fig. 6.

(1) Baseline Model: For the baseline model, a convolutional neural network was chosen. It is consists of 2 convolutional layers: the first one has kernel size 4, 32 channels;and the second one has kernel size 4, 16 channels. Both layers have no padding and stride of 1, followed by a batchnorm layer, a dropout layer, and a maxpooling layer. Activation function used is ReLU. Then, flatten layer and 2 fully connected layers were used to adjust the dimension of the output logits from 4624 to 128, finally to 15, matching the number of classes. Only the image was feed into this network. In other words, this model is a end-to-end model with images as input and classification as output.

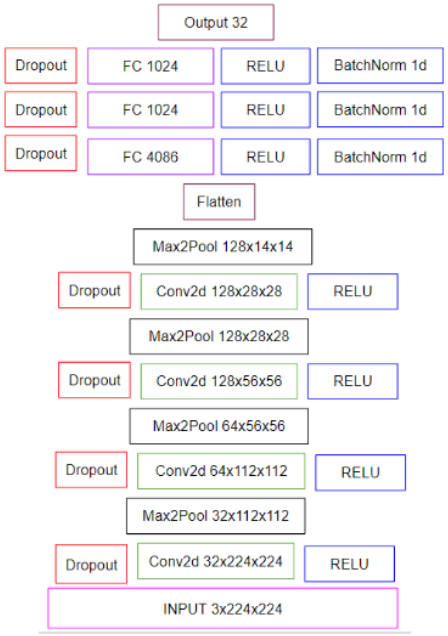

Figure 3. CDRP architecture.

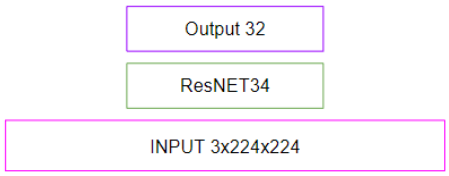

Figure 4. ResNET34 architecture.

(2) Image-Joints Model: Two ways of incorporating the joint localization information were implemented. Both methods shares one core idea: combining the image features and the joints features for better classification performances. For consistency purpose, the image feature extractor has the similar structure as the baseline classifier. It used exactly the same 2 convolutional layers with batchnorm layer, dropout layer, and maxpooling layer as the baseline model. Instead of mapping the flattened image features directly into dimension 15 for classes, it is adjusted to dimension of 128 as image features.

The joints feature are a total of 16 keypoints with their respective x, y locations on the image that locates the 16 body parts of a person. Invisible keypoints indicated by the flag were represented by a placeholder of zeros. Let $\mathbf{x}_i$ be

the 2-dimensional character vector for the i-th keypoint in an image. A single person's joints feature of length n is represented as

$$\mathbf{x}_{1:n} = [\mathbf{x}_1; \mathbf{x}_2; \ldots; \mathbf{x}_n] \tag{2}$$

For the first version of joint feature extractor, the team picked a naive fully connected MLP. The joints feature is flattened such that the input dimension is set to 32. Then, linear layers changes the feature dimensions from 32 to 64, then to 128, with a 1D batchnorm layer in between.

The second idea of embedding the joints feature stems from the character-level embedding layer in NLP [5]. Because the joints feature only contains information of single keypoints, 1D-convnets can be used to connect the different keypoints to enable the model to have a better understanding of the person's posture. The architecture of the model is similar to one shown in 5. The figure shows 2-channel convnet embedding on the word level in NLP, whereas our model applies a one-channel convnet on joints. A convolution operation uses a filter $\mathbf{w} \in \mathbb{R}^{hk}$ which is applied to a window of h joints to produce a new feature. Then, a feature c_i is generated from a window of characters $x_{i:i+h-1}$ by

$$c_i = f(\mathbf{w} \cdot \mathbf{x}_{i:i+h-1} + b) \tag{3}$$

where b is a bias term and f is a non-linear function, which we used ReLU. The filter is applied to each window of joints to produce a feature map $\mathbf{c} = \{c_1, c_2, \ldots, c_{n-h+1}\}$. Then, a maxpool is applied over the feature map and take the maximum value $\hat{c} = max\{\mathbf{c}\}$ as the feature corresponding to this filter. The joints feature embedding layer contains two 1D-convnets followed by a single linear layer to produce a hidden layer of size 128. The window size we used for this model was 3.

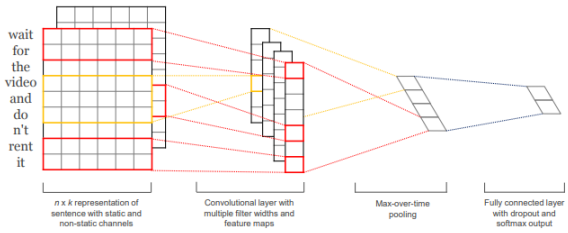

Figure 5. Sample convnet architecture. Image is taken from [5].

With the images features extracted and the joints features extracted in either version, a large feature tensor was concatenated with both features $\mathbf{h}_i = [\mathbf{h}_{imgi}; \mathbf{h}_{ji}]$. The dimensions of the concatenated features was 256. Then, the first 2 linear layers both have input and output of dimension 256. Finally, the third layer maps the 256 dimensions into dimension of 15 for classification.

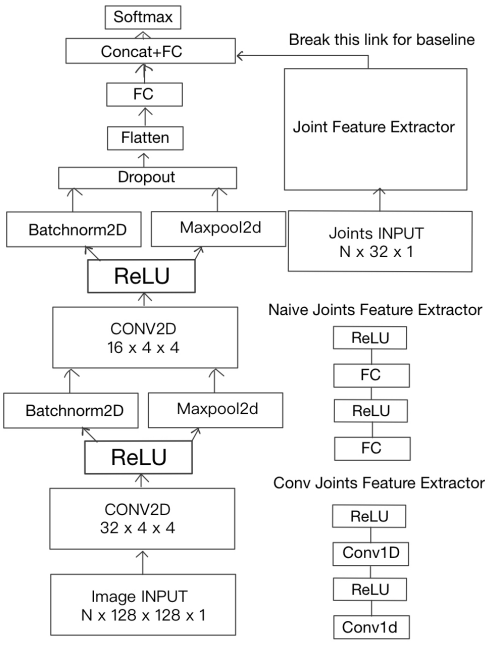

Figure 6. Classifier Models Structure

2.3. Metric

The metric we will be using consist of quantitative and qualitative results.

For the intermediate joint estimation, we will be monitoring upon the training and validation loss evolution mainly, as the regression loss reflects how close our labelled points are to the ground truth. After training, we will then be applying PCKh@0.5 metric, a accuracy checking metric for human pose estimation, on the test set. PCKh@0.5 is a threshold value based on half of the head length calculated from the ground truth joint labels. The distances between the ground truth and the predicted score for seven subparts of the body (Head, Shoulder, Elbow, Wrist, Hip, Knee, Ankle) are calculated. If the distance fall within the threshold, it is deemed as accurate. The mean accuracy of all subparts will also be calculated. Qualitatively, we would visualize the test images with key-points plotted on them.

For action classification, in addition to the loss function, classification accuracy was used as metric. This accuracy simply calculates the correct classified counts for a batch divided by the total number of data in a batch. Same accuracy was used for both training and validation.

2.4. Experiment Details

For the baseline localization model, we used a batch size of 128 on over 9500 training data. A Nestorov Momentum of 0.9 was used as optimization. A learning rate of 0.0005 with no decay was applied and no regularization was used.

We used a dropout prob of 0.1. The model was trained for 270 epochs as both loss flattened out.

For the ResNET localization model, we used the same setup as the baseline method. It was also trained for 180 epochs as both loss flattened out.

The benchmark localization model is trained for 2 days on a learning rate of 0.00025.

For classifier models, different combinations of hyperparameters were tried, while only the ones with best performances will be reported.

For the baseline classifier model, batch size was set to be 128. Adam classifier was used for optimization. It used the learning rate of 0.0001 with no decay, and l2 weight decay with strength of 0.005 for regularization purpose. It is trained for 50 epochs.

For the classifier model with naive joint features extractor, batch size was also set to be 128. Similarly, Adam classifier was used for optimization. It used the learning rate of 0.0002 with no decay, and l2 weight decay with strength of 0.001 for regularization purpose. It is trained for 36 epochs.

The image-joints classifier model used similar hyperparameters as the baseline model, except that the learning rate was set to 0.001 and later decayed to 0.0001 after the 10th epoch.

3. Results

3.1. Joint Localization Models Results

For the baseline and ResNET34 model for localization, the training loss curve is shown and compared in Fig.7. The validation loss curve is shown and compared in 8. The visualization results from CDRP model and ResNET34 model are shown in Fig. 9 and Fig. 10.

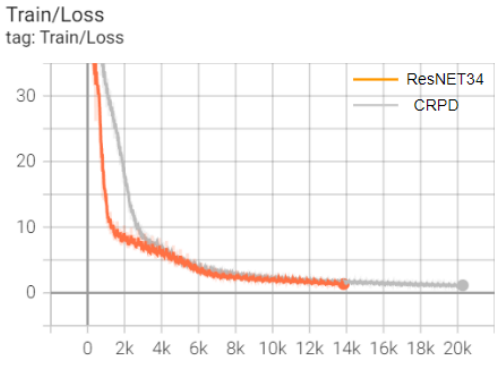

Figure 7. Training Loss of CDRP and ResNET34 model

We compared the performance of our method to the benchmark model, which has a training history presented in Fig. 11, while the visualization is presented in 12.

We used PCKh@0.5 criteria to measure the accuracy of the predicted joint estimation on the test set. The result is listed in Table. 1.

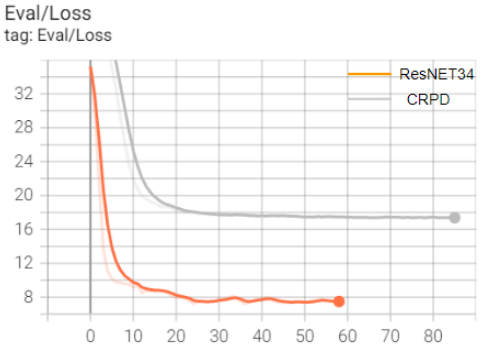

Figure 8. Training Loss of CDRP and ResNET34 model

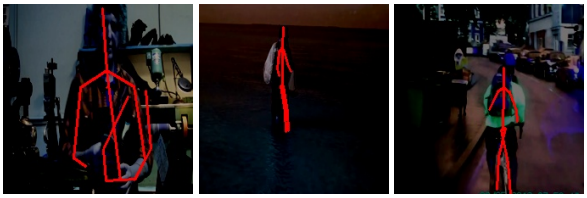

Figure 9. Joint visualization on CDRP model at test time.

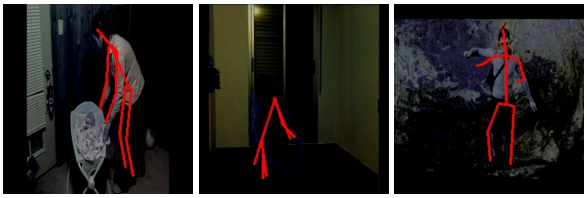

Figure 10. Joint visualization on ResNET34 model at test time.

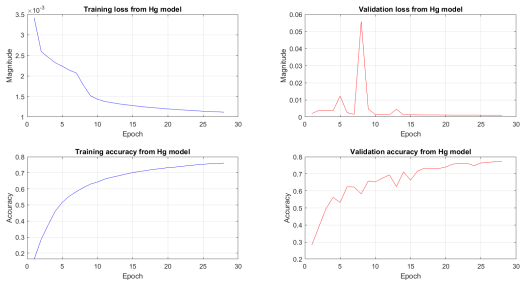

Figure 11. Training Loss, validation loss, training accuracy at PCKh@0.5 and validation accuracy at PCKh@0.5 on Hourglass mode.

3.2. Activity Classification Models Results

For the baseline model for classifier, the training loss and accuracy curve is shown in Fig. 13 and 14. The validation accuracy curve is shown in Fig. 15.

Figure 12. Joint visualization on Hourglass model at test time.

Part/Model	CDRP	ResNET34	Hourglass
Head	23.84	30.46	95.46
Shoulder	36.98	43.83	90.06
Elbow	30.30	37.91	81.44
Wrist	21.18	27.52	75.40
Hip	31.71	39.44	80.82
Knee	24.10	34.02	75.54
Ankle	18.21	26.99	70.67
Mean	26.00	34.37	81.46

Table 1. PCKh@0.5 Joint Accuracy Results for the bes CDRP, ResNET and Stacked Hourglass during training.

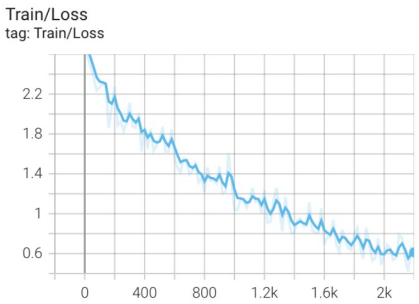

Figure 13. Training Loss vs. Iterations for Baseline Classifier

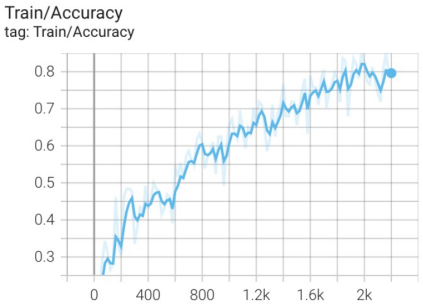

Figure 14. Training Accuracy vs. Iterations for Baseline Classifier

For the classifier model with naive joints feature extractor, the training loss and accuracy curve is shown in Fig. 16

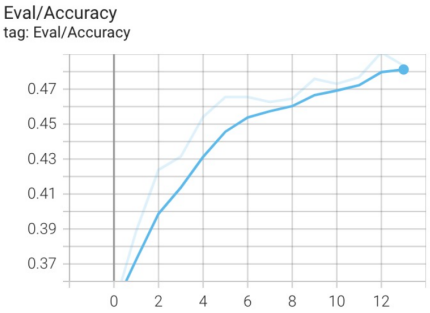

Figure 15. Validation Accuracy vs. Iterations for Baseline Classifier

and 17. The validation accuracy curve is shown in Fig. 18.

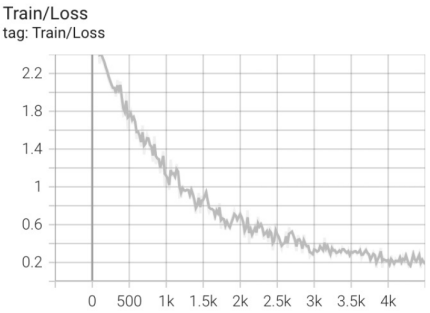

Figure 16. Training Loss vs. Iterations for Classifier with Naive Joints Feature Extractor

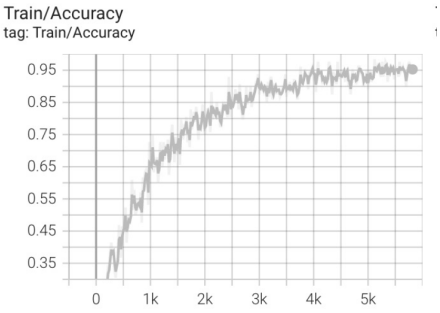

Figure 17. Training Accuracy vs. Iterations for Classifier with Naive Joints Feature Extractor

For the classifier model with convolutional joints feature extractor, the training loss and accuracy curve is shown in Fig. 19 and 20. The validation accuracy curve is shown in Fig. 21.

For numerical comparison, different models' losses and accuracies are reported in Table. 2.

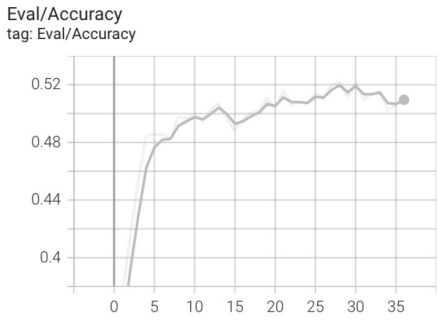

Figure 18. Validation Accuracy vs. Iterations Classifier with Naive Joints Feature Extractor

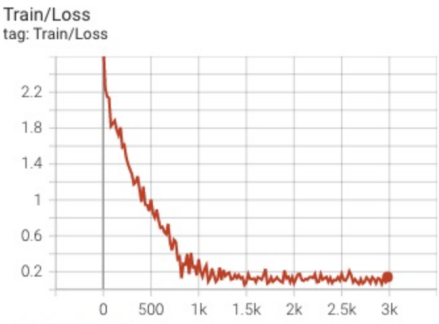

Figure 19. Training Loss vs. Iterations for Classifier with Convolutional Joints Feature Extractor

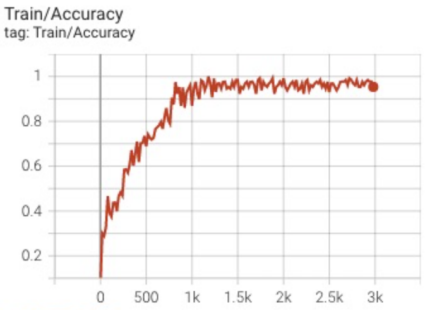

Figure 20. Training Accuracy vs. Iterations for Classifier with Convolutional Joints Feature Extractor

4. Discussion

Comparing the two localization models we implemented, both baseline and ResNET34 models with regression loss converges to a training loss as low as 1.0 on the training data. This indicates the learned model can confidently label the keypoints for a trained image. With the same training parameters set up, the ResNET34 model converges faster then the CDRP model because the main body of the network besides the final fc layer was pre-trained, so the whole model would generally provide a faster convergence rate even the network itself is deeper. The evaluation

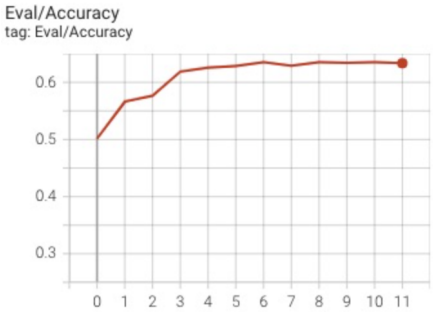

Figure 21. Validation Accuracy vs. Iterations Classifier with Convolutional Joints Feature Extractor

Model	Train Loss	Train Acc	Val Loss	Val Acc
Baseline	0.542	0.8321	1.869	0.4834
LFE	0.1522	0.9531	1.715	0.5219
CFE	0.1419	1.0	1.458	0.6353

Table 2. Results Comparison for Different Classifier Models: Baseline, Linear Feature Extractor on Joints input, and Convnet Feature Extractor on Joints input

loss for CDRP converges to around 18 and for ResNET34 it converges to around 7. This shows with a appropriately deeper neural network structure, the joint estimation becomes more accurate and close to the ground truth because more human features could be captured.

We then compared these two methods with our benchmark model. From the PCKh@0.5 table in Table. 1, we see that even though ResNET34 model does better on the localization accuracy than baseline method, it is still very far from the benchmark standard who reached a mean accuracy of 81.46. This is because the method we were using was building a single CNN network that trained with L2 loss, and more advanced methods such as the stacked hourglass model utilized more complex model structure and probability-based loss or heatmaps, providing a more stable and accurate result while dealing with cases where certain joints are being labelled as invisible.

However, we believed our model worked well in majority of test cases where there are little occlusion and the front or side body of the human is visible. The test image with joint estimation result, as shown in both Fig. 9 and Fig. 10, demonstrates the ability for our model to capture the joint locations of the target person. We noticed that the generated joints may not be at the precise locations as ground truth, but the body skeleton that connects the joints is the right indication of what the person is doing. This is helpful as it provides an extra information for the downstream model to acknowledge.

For the baseline classifier, the team tried more than 20 combinations of hyperparameters and different dataset splits. Even though the training accuracy was successfully trained to over 80 percent, the validation accuracy only reached 48 percent. Based on the learning curves, error analysis were done. As a result, the team observed that for baseline model, over 30 percent of the mislabeled data during validation are for those categories that have relative less data. Such data imbalance issue was discovered with 20 classes classifier output. Even after the 5 least images categories were removed, similar situation still occurred. Besides, the dataset itself was considered challenging as the human accuracy for classification of the dataset was about 80% from the results of our team. Another problem from our dataset was the tendency of overfitting. Although there are over 25,000 images in the MPII dataset, most of them either did not contain joint location information or had too few joints marked. After filtering those images, only 11,000 images were suitable for training. With such a small dataset size and 15 classes to classify, our model capacity needs to be greatly limited as well to alleviate the overfitting issue. However, even with reduced model complexity, overfitting still occured, as can be seen from the results plots shown above. Other possible reasons of overfitting could be, firstly, human body only takes a small portion of the whole images for many data. This could lead to the situation that the model overfits the background information easily. For example, white backgrounds are easily classified to "winter activity" label.

The team expected the addition of joints feature extractor to increase the performances by emphasizing the human gesture manually. After the naive linear joints feature extractor being implemented, the validation loss increased slightly to 52.19 percent. This showed that by simply adding the keypoint information was not significant in improving model performance, as the more complex implications of human gesture was not captured by the linear joints embedding layer. With the convnet embedding layer, the joints connection information is preserved, and the classification model experienced a 20% increase in accuracy compared to baseline.

5. Future Work

One of the most promising possible improvement to this project in changing dataset. For the MPII dataset [2], we discovered that not only missing label situations occurred a lot, data imbalance issue were also apparent. Due to time constraint, the team couldn't explore any other available dataset that works for both upstream and downstream models. Another option would be manually adding joint location labels to the dataset to increase the available data size.

For the localization model, because of the time constraint of the project, there can still be great room to improve on our existing model including fine-tuning different parameters. The more important target for the team is to investigate and re-implement some benchmark models, and see whether we could improve from there. This ensures the predicted joint estimation is reliable when fed into the downstream model.

As for the classification model, we have seen that adding human posture information to the image would increase the human action classification accuracy. However, the joints embedding layer only consisted of convnets and linear layers. With a total of 16 joint keypoints, it is possible for the joints model to contain more information about the action and posture of a person, and that the model could self-identify which keypoint or joint connections should be attended to during posture identification. Hence, the Transformer model [9] may be useful in achieving such task. With self-attention in the Transformer model, joints and posture context information might be better embedded and is possible to achieve better classification results. However, as discussed above, because of the overfitting issue from the lack of training data, the Transformer model's capacity would almost certainly lead to worse overfitting. This should be considered only if the dataset size is increased.

References

[1] Mahmoud Al-Faris, John Chiverton, David Ndzi, and Ahmed Isam Ahmed. A review on computer vision-based methods for human action recognition. *Journal of imaging vol. 6,6 46*, 2020. 1

[2] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 1, 2, 8

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2

[4] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *European conference on computer vision*, pages 34–50. Springer, 2016. 2

[5] Yoon Kim. Convolutional neural networks for sentence classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1746–1751*, 2014. 4

[6] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016. 2

[7] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V. Gehler, and Bernt Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2

[8] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014. 1, 3

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 8

[10] Yang Wei. pytorch-pose. `https://github.com/bearpaw/pytorch-pose`, 2019. 3

[11] Weilong Yang, Yang Wang, and Greg Mori. Recognizing human actions from still images with latent poses. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2030–2037, 2010. 2