Human Pose Estimation Using a Stacked Hourglass Network

Robert Lee
Electrical and
Computer Engineering
University of Victoria
rklee@uvic.ca

Julian Rocha
Software Engineering
University of Victoria
julianrocha@uvic.ca

Wanze Zhang
Computer Science
and Statistics
University of Victoria
wanzezhang@uvic.ca

Nicole Peverley
Software Engineering
University of Victoria
nicolepeverley@uvic.ca

Rafay Chaudhry
Software Engineering
University of Victoria
rafaych@uvic.ca

Corey Koelewyn
Software Engineering
University of Victoria
ckoelewyn@uvic.ca

Abstract

Human pose estimation (HPE) is the task of identifying body keypoints on an input image to construct a body model. The motivation for this topic was driven by the exciting applications of HPE: pedestrian behaviour detection, sign language translation, animation and film, security systems, sports science, and many others. HPE shares many challenges with typical computer vision problems, such as intra-class variations, lighting, perspective, and object occlusions. It also faces challenges unique to HPE such as strong articulations, small and barely visible joints, and self-occlusions from overlapping joints. This report discusses a stacked hourglass network architecture that was developed and trained from scratch to achieve performance comparable with models on the COCO leaderboard from late 2016. This work uses the existing COCO 2017 Keypoint Detection dataset. The final model performs very well on most images, especially those containing well-separated people with the subject centered in frame. It struggles with images containing highly overlapped people or heavily occluded or articulated keypoints.

1 Introduction

Human pose estimation (HPE) is defined as the problem of estimating joint keypoints on an input image to visualize a model of the body's pose. Since HPE can be done for the whole body or a part of the body, the applications are widespread and benefits many industries. For example, HPE can be used to estimate joints in a hand, the application of which extends to computer interpretation of sign language. Full body estimation can be used for animation, security systems, sports science, and much more.

Our group created a stacked hourglass network that was trained on the Common Objects in Context (COCO) dataset. We focused on estimating a maximum of 17 keypoints spanning the full human body on a 2D image. A number of challenges make HPE a difficult problem domain; these challenges include variability in human appearance and physique, environment lighting and weather, occlusions from other objects, self-occlusions from overlapping joints, complexity of movements of the human skeleton, and the inherent loss of information with a 2D image input. This largely unsolved problem enabled us to explore many novel and creative approaches, enriching our learning experience. We are excited to report the results we yielded.

2 Related Work

Prior to the advent of deep learning techniques, HPE approaches have relied on traditional computer vision techniques such as Histogram of Oriented Gradients or deformable part models; however, these models had difficulty with multi-person pose estimation and occluded keypoints. The first significant deep learning model for HPE was "DeepPose: Human Pose Estimation via Deep Neural Networks" (2014). DeepPose used a convolutional neural network (CNN) to solve a regression problem on joint coordinates. They also used a cascade of regressors to improve their estimation. DeepPose achieved an average score of 0.61 for Percentage of Correct Parts (PCP) [1]. A 2015 model, "Efficient Object Localization Using Convolutional Networks", used a CNN with a cascading architecture that combines a fine and coarse scale convolution networks [2]. This model chose to predict keypoints by using heatmaps. This 2015 model achieved a score of 82.0 for the full body PCK @ 0.5 [2]. An altogether different approach was taken by the 2016 "Iterative Error Feedback (IEF)" model. IEF tries to predict keypoints by using a self-correcting model that incrementally changes an initial solution. Incremental changes are made based on feeding back error predictions. IEF achieved a score of 81.3 on full body for PCK @ 0.5 [3]. Lastly, a model our group took inspiration from is "Stacked Hourglass Networks for Human Pose Estimation" from 2016. The stacked hourglass networks allow for repeated bottom up (high to low resolution) and top down (low to high resolution) inference. This network outputs a series of heatmaps (one for each joint). The Stacked Hourglass Network achieved an impressive score of 90.9 on full body for PCK @ 0.5 [4].

3 Method

This section begins by outlining the problem formulation. Then, the specifics of the implemented data pipeline, model architecture, and model evaluation metrics are described.

3.1 Problem Formulation

HPE systems can be categorized into 2D vs 3D and single-person vs multi-person. To improve the feasibility of our project, we have focused on single-frame single-person monocular RGB images. Current state-of-the-art techniques for 2D single-person HPE can be categorized into two categories: regression on absolute joint position, or detection on joint locations with heat maps. Since a direct mapping from the input space to joint coordinates is a highly non-linear problem, heat-map-based approaches have proven to be more robust by including small-region information [5]. Thus, we have chosen the heat map approach.

There are three different types of models used with full body HPE: kinematic, contour, and volumetric, as shown in Fig. 7 [5]. A kinematic model resembles a stick-figure skeleton. The contour model consists of 2D squares and rectangles that represent the body, and the volumetric model represents the body with 3D cylinders. The kinematic model is the simplest model to perform loss metric computations, and thus is preferred by our group as a scope-limiting decision to simplify the problem space. Our goal is to predict a kinematic model for the individual in each picture.

We chose to use the COCO Keypoint dataset [6]. This dataset consists of 330 K images, of which 200 K are labelled. There are pre-sorted subsets of this dataset specific for HPE competitions: COCO16 and COCO17. These contain 147 K images labelled with bounding boxes, joint locations, and human body segmentation masks. We originally considered using DensePose [7], which is a highly detailed manually annotated subset of the COCO dataset, but found it does not offer joint coordinate labels. Another popular dataset is the MPII dataset [8], which consists of 41 K labelled images split into 29 K train and 12 K test. We originally planned to use this for validating our model's performance.

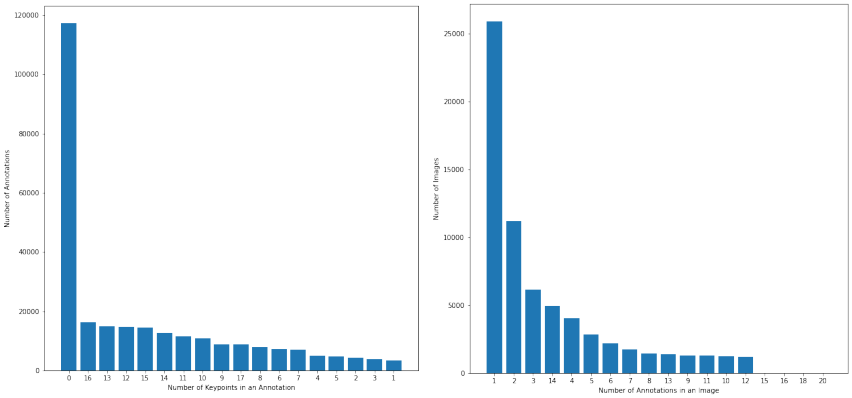
3.2 Data Pipeline

The COCO dataset has more than 20 GB of images so keeping the entire dataset in memory during training is not an option. Image pre-processing needs to be done to get the images and annotations in a format that can be passed to the model. A data generator was developed to tackle these two tasks. To prevent the generator from being the bottleneck of the training process, the data generator runs on CPU concurrent with the GPU training. The generator fetches images from disk in batches and

the next batch can be fetched and processed while the model is performing the forward and back propagation on the current batch.

3.2.1 Data Filtering

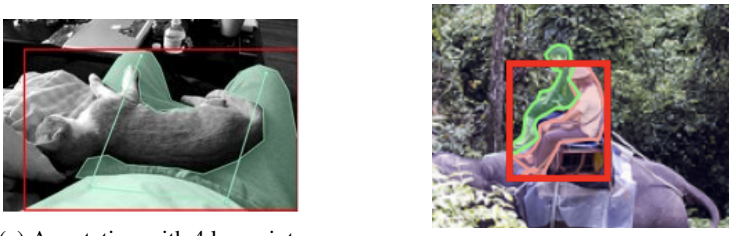
There are 66,808 images in the COCO dataset containing a total of 273,469 annotations. As shown in Fig. 1 a), most of the annotations in the COCO dataset do not have all 17 keypoints of the body labelled. The model should not expect a perfect human image with all keypoints visible in frame. The model should instead output a dynamic number of keypoints based on what it can find. But what is the purpose of an annotation with 0 to 4 labelled keypoints? Fig. 2 shows examples of these annotations with few labelled keypoints. Clearly the bounding boxes of these annotations denote people, but because there are not many keypoints, these examples may confuse the model. If the model should be shown examples with 0 keypoints, then images that do not contain people would be more helpful. 5 keypoints was chosen intuitively as the minimum number of keypoints for a usable example; any less and the image rarely contains enough information to clearly make out a person. Therefore, despite the fact that 0-4 keypoint annotations make up 48.86% of the total COCO dataset annotations, these annotations were filtered out during training.



(a) Number of annotations with a given number of keypoints

(b) Number of images with a given number of annotations

Figure 1: COCO metrics



(a) Annotation with 4 keypoints

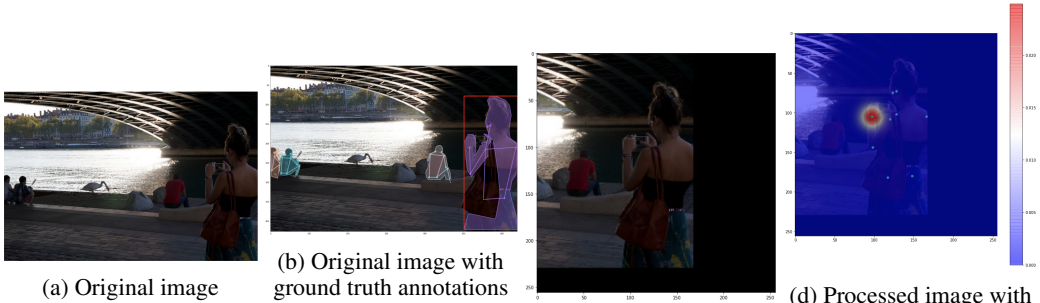
(b) Annotation with 0 keypoints

Figure 2: Examples of annotations with 0-4 labelled keypoints

3.2.2 Data Pre-processing

Even though our goal is a model that estimates the pose of a single person in the image, 61.28% of the COCO images contain more than one annotated person. The annotations per image are broken down in Fig. 1 b). It would be desirable if multi person images did not need to be discarded, so cropping to a bounding box converts a multi person image into a single person image. Keeping these images with multiple people has many benefits. The main benefit is training the model to label the person in the direct center of the image, in cases where multiple people (and thus multiple joints) are present. This is a more realistic use of the model, as it is unlikely that real-world images are always single-person. The other benefit is training on a much larger dataset.

The pre-processing responsibilities of the data generator include: cropping to the ground truth bounding box of a person, resizing to the models input resolution and dimensions, performing random data augmentation, and converting ground truth annotations for each keypoint to a Gaussian heatmap for the cropped images. Fig. 3 shows an example of the transformations. Fig. 3 only shows the cropping for one person but since there are 4 annotated people, the image would get split into 4 images, each centered on the person of interest. Fig. 3 also only shows the heat map of the left hand, but since COCO annotations contain 17 keypoints, it produces 17 heatmaps per annotation.



(a) Original image

(b) Original image with ground truth annotations overlaid

(c) Processed image

(d) Processed image with translated keypoints and left hand heat map overlaid

Figure 3: Example of transformations applied by the data generator

3.2.3 Data Augmentation

Having a model that could generalize was important to us. We decided using data augmentation to transform the images and annotations would help us achieve this. The different types of transformations applied to the images for data augmentation were:

- Blurring an image makes the colour transitions between objects in an image less sudden. This is helps improve performance for objects which may be out of focus.
- Sharpening an image makes the edges in the image more defined.
- Sharpening blending factor determines how much variation is tolerated when sharpening an image.
- Hue and Saturation determines the dominant colour and its intensity. This helps build robustness to colour and appearance, which is important because of the large variability in clothing.
- Brightness determines the average pixel values for an image. This helps build robustness in extreme lighting conditions, such as when the subject only has a silhouette visible, or is overexposed.
- Contrast is the range of pixel values in an image. Higher contrast images will have a wider range of pixel brightness values. This improves high contrast situations such as sunsets.
- Pixel dropout masks out the original value of a pixel, forcing the model to be more robust with noisy images. This improves performance by forcing the network to have redundancies built in.
- Gaussian noise is similar to Pixel dropout, but instead of masking a value it is changed by an amount randomized by a Gaussian process. This improves noise tolerance as well.
- Scaling increases or decreases the size of the image as the model sees it. For the cases where keypoints would no longer be inside the bounding box, those keypoints would be removed. This improves scale invariance.
- Rotation would rotate the image around the center by a certain number of degrees. This improves invariance to the orientation of the person.
- Right-left flip is the probability that the image would be mirrored. This is especially important because the model should be agnostic to the direction a person is facing.

Due to the large number of parameters we were introducing into training with data augmentation, we decided to have 3 levels of augmentation. We introduced heavy, medium and light parameters which are specified as command line arguments, which would determine the values for each of the augmentation metrics. The values for this can be seen in Table 1.

Table 1: Range of values for different augmentation levels

Augmentation Metric	Heavy	Medium	Light
Blur with probability 30% (sigma)	2.0	1.5	1.0
Sharpening	0.8 to 1.2	0.85 to 1.15	None
Sharpening blending factor (between)	(0, 1)	(0, 0.5)	None
Hue and Saturation change (out of 255)	-30 to 30	-20 to 20	-15 to 15
Brightness (out of 255)	-30 to 30	-25 to 25	-20 to 20
Contrast	0.75 to 1.25	0.85 to 1.15	0.90 to 1.10
Pixel Dropout (% of pixels)	10	5	3
Gaussian noise (out of 255)	5%	3%	1%
Scaling (+/- %)	25	20	15
Rotation (+/- degrees)	30	25	15
Right/left horizontal flip (probability)	0.5	0.5	0.5

Not all augmentations were applied to each image, with certain augmentations being exclusive of each other. Brightness and contrast were never applied to the same image due to both having an effect on pixel values. Similarly, Gaussian noise and pixel dropout overlap with introducing noise, so we opted to never apply them at the same time. Image crop and rotation were also not applied to the same image, as it removes elements of the image which are out of frame after each operation. Applying a rotation followed by a scale down results in keypoints which correspond to removed areas of the image being re-introduced. This would confuse the model by assigning a joint label to an empty area of the image.

3.3 Model Architecture

We debated whether to structure the model architecture to perform a regression on joint coordinates, or a detection-based approach that used confidence maps for each joint. Since this problem domain was highly non-linear, recent works have found that a heatmap-based approach performed much better than regression because of the ability to include local pixel confidence information in cases where joints are not obvious (see Table 3). We chose a refinement of the stacked hourglass network first introduced by *Newell* in 2016 [4]. Variations of this network have proven extremely popular in HPE single-person 2D literature. Seven of twelve notable detection-based works shown in Table 3 use this building block.

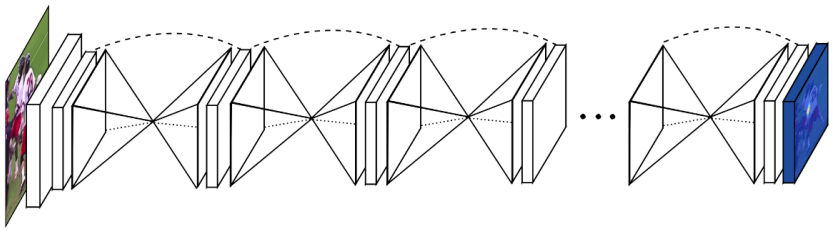


Figure 4: A stacked hourglass network for HPE [4]

The network, shown in Fig. 4, consists of a series of stacked U-Nets [9]. The network gets its name because the U-Nets resemble hourglass structures. Each U-Net, shown in Fig. 5 is a lightweight encoder-decoder structure that consists of residual blocks with either convolution or upsampling applied for the encoder and decoder stages, respectively. Unlike typical U-Nets that have proven successful for problem domains such as semantic segmentation, this network does not use unpooling or deconvolutional layers during the decoder stage. Instead, nearest neighbour upsampling is used. Skip connections link feature levels of the same spatial resolution in the encoder and decoder stages.

This structure allows the network to combine information from deep abstract features, and local high-resolution information.

The input to the entire model is a RGB image of resolution 256x256. Since performing operations at original resolution is expensive in compute and memory, all the internal hourglass stacks have a max resolution of 64x64. Thus, between the input layer and the first hourglass block, the input resolution is brought down from 256x256 to 64x64 by using the following operations: a 7x7 convolutional layer with stride 2, a residual module, and max pooling.

The input to each hourglass block is at a resolution of 64x64. The encoder block performs top-down processing, where the image spatial domain is decreased while increasing feature depth. After each convolution block, max pooling is applied to reduce in spatial size. The network also branches off at the pre-pooled resolution to apply more convolutions to form the *skip connections*. At the smallest stage in the middle of the hourglass, which is denoted the *bottleneck*, the network is at a resolution of 4x4 pixels. Here, the convolution operations can compare global features in the image. To reconstruct the original resolution, the network applies nearest-neighbour upsampling and joins feature information from the skip connections. The final resolution is identical to the input at 64x64, and the network is fully symmetric.

At the output resolution, the network predictions are constructed by applying two rounds of 1x1 convolutions to the extracted features. This forms a series of 17 one-channel heatmaps, one for each joint, where the intensity of the pixel value corresponds to the probability that a joint is found at that location. This intermediate prediction is added with the feature map and used as input to the next layer. In essence, each block in this architecture performs a refinement of predictions generated by the previous block. The number of hourglass blocks does not affect the output resolution, since each block is symmetric.

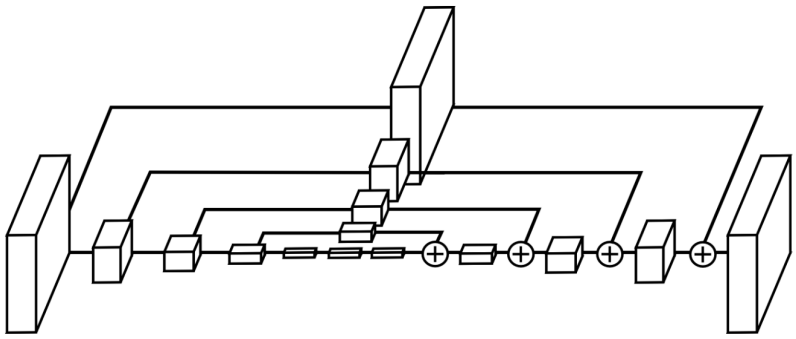


Figure 5: A single hourglass module. Each block consists of a residual module. [4]

3.3.1 Intermediate Supervision

Deep networks can often suffer from vanishing gradients, which is where the gradients of the loss function approach zero deep into the network. This is often due to the activation functions, such as sigmoid, having a range of values where the gradient is extremely small. Since these gradients multiplied together using the chain rule during backpropagation, this can cause the gradient to disappear deep into the networks. These gradients are used to update the weights during backpropagation, so vanishing gradients can stall learning and result in a poorly performing network.

To mitigate this problem, the network architecture uses both residual blocks and intermediate supervision. Residual blocks adds both the modified output from the convolution and activation operations, and the original values. This permits an alternative path for gradients to flow through the network. Since this network is highly symmetric, intermediate supervision was used as well. The ideal output for a perfect network would have each hourglass block output identical heatmaps, we extract intermediate prediction heatmaps to perform loss function evaluations. This allows the network to re-introduce gradients deep into the network, reducing the risk of vanishing gradients.

3.3.2 Other Model Parameters

The network uses a mean-squared error loss function to evaluate the prediction heatmaps against the ground truth heatmaps. The equation is shown below in Eq. 1. The learning rate was scheduled between 5e-3 and 5e-4, with the decrease occurring at around epoch 70. We originally used the `rmsProp` optimizer, but we discovered that the `Adam` optimizer stabilized our validation loss.

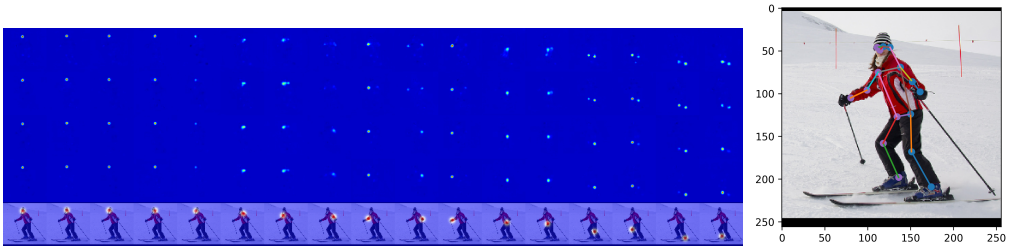
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{1}$$

3.4 Model Evaluation

Since HPE is a complex problem with many ways to measure success, it was not sufficient to simply report the loss and accuracy of the model. Visual insight into what different model versions see would allow various issues/bugs to be more easily identified. For this reason, methods to visualize heatmaps and predicted keypoints were developed. Furthermore, insight into the precision of keypoint predictions across varying thresholds were needed to to determine when predictions were relatively close, but not perfect matches, to the ground truth. For this reason, the Object Keypoint Similarity (OKS) metric was developed. Finally, insight into performance on specific joints was revealed through the implementation of the Percentage of Correct Key Points (PCK) metric. These three evaluation methods helped immensely during hyperparameter tuning.

3.4.1 Visualization

To gain insight into our model's predictions, we first visualized and compared the 17 output heatmaps, corresponding to COCO joints, from each successive hourglass layer with the image's ground truth heatmaps. This is shown for a 4 layer hourglass model in Fig. 6 a). This figure shows the architecture refinement in the top right corner, where the model originally predicts two points on the heatmap with approximately equal brightness until gaining more confidence on one point in the final hourglass layer. Often, this refinement would help distinguish the models confusion between the left and right of each joint. In order to evaluate the model, we converted the predicted heatmaps back into COCO formatted keypoints. This conversion was accomplished by first upscaling each heatmap from 64x64 to the original image size of 256x256, then using a gaussian filter to blur each heatmap, and finally choosing the maximum point in the heatmap with non-maximum suppression by forcing values less than the determined threshold of 0.04 to 0. One set of keypoint coordinates is determined for each heatmap and later used for evaluating the model. For qualitative assessment of the model, a method to visualize the estimated keypoints in a skeleton overlayed on the input image was implemented, shown in Fig. 6 b).



(a) Visualization of Heatmap Outputs from Epoch 15

(b) Keypoint Skeleton Prediction

Figure 6: Model visualizations

3.4.2 Evaluation Metrics

The goal of this project was to develop an HPE model that can perform with high accuracy and generalize well to unseen data. The success of the model was measured using 2 quantifiable metrics common to the HPE literature [10].

The first metric that was implemented for evaluation was OKS. We computed this metric across epochs to determine our models with the highest accuracy. Each model has rapid improvement in the first 5 epochs and further epochs have slower and more gradual improvement as seen in Fig. 8. Our highest performing model was able to achieve an OKS primary challenge metric score of 0.575 and an OKS loose metric score of 0.795. This score is competitive with models on the COCO leaderboard from 2016. Using horizontally flipped images and taking the average bumped the scores by 3-5% for this metric. OKS is commonly reported in the literature in terms of AR (average recall) and AP (average precision). It was implemented using the COCO Python API [11]. The API allows for evaluation of results and the ability to compute precision and recall of OKS across scales. It required that our model outputs be formatted according to the COCO keypoint detection standard [12]. Beyond model evaluation, the API also provided methods for detailed analysis of errors with plots that were explored and aided in parameter tuning.

The second metric was PCK [13] which we implemented ourselves, separately from the COCO evaluation API. There are a number of variations of PCK, and from our research it does not appear to be a standardized metric. PCK considers a detected joint as correct if the distance between the predicted and the true joint is within a specified threshold. We implemented a variation of PCK@0.2, which uses a threshold of 20% of the torso diameter from the ground truth keypoints. Since the literature is not clear on the metric's behaviour when one or both hip points are not present, we implemented a secondary measure of 20% of the head diameter. Our default case is an empirically determined average hip width from the dataset if neither a torso or head was detected in the image. Our highest performing model achieved 0.787 on average for each joint PCK. The model PCK over epochs is graphed in Fig. 11. The results broken down for each joint are specified in Table 2.

Table 2: PCK breakdown by joint for Epoch 107, Hourglass 4

Joint	PCK Score (percent)
nose	0.8829
left eye	0.8842
right eye	0.8900
left ear	0.8273
right ear	0.8320
left shoulder	0.7289
right shoulder	0.7290
left elbow	0.7636
right elbow	0.7721
left wrist	0.7517
right wrist	0.7544
left hip	0.6526
right hip	0.6478
left knee	0.8014
right knee	0.8056
left ankle	0.8290
right ankle	0.8289
average	0.7872

4 Experiments

Here are a list of experiment examples we have tried in order to improve our model's performance.

4.1 Using Full Training Set vs Subset

In order to save time when training and evaluating, we hand picked 10 images to compile a representative set of different types of input images our model was likely to encounter. We also created a parameter to subset the data into different sizes. This allowed us to test out different hyperparameters more quickly and have a better idea if we wanted to pursue longer training with specific tuning parameters. A smaller sample size also allowed us to test our evaluation process to ensure it was working as expected before allocating time to run evaluations and plot graphs.

4.2 Hourglass Model with 4 Layers vs 8 Layers

As mentioned above, our dataset was fully trained on a 4-layer hourglass model (hourglass 4) and obtained a loose OKS metric score of 0.795. We also started experimenting 8-layer hourglass model (hourglass 8). Even though we have not fully trained our data with hourglass 8, it is expected that hourglass 8 will lead to a higher accuracy. Comparing OKS score graphs for hourglass 4 and hourglass 8 (Fig. 10 vs Fig. 8), we can see that it took more time to train in the beginning with the hourglass 8 model, but the OKS score had already reached hourglass 4's best performance around epoch 70. Another strong indicator to hourglass 8's potential was with heavily overlaid people. This is shown when comparing 4-layer and 8-layer heat maps (Fig. 12, Fig. 13, Fig. 14, Fig. 15). The skeletons produced by the 8-layer models show a clear distinction between the two people (Fig. 16, Fig. 17).

4.3 With Data Augmentation vs Without Data Augmentation

Data Augmentation was one of the most vital experiments we performed. Augmenting data can help a model generalize and combat overfitting. We ran a series of transformations, most of which were covered in the section 'Data Augmentation'. However, for convenience the data augmentations we applied were: blurring, sharpening, hue and saturation, brightness, contrast, pixel dropout, Gaussian noise, scaling, rotation and flips across the y-axis. As seen in Fig. 21, data augmentation improved results by 5%.

4.4 Sigmoid vs Linear vs Relu Activation

Different activation functions were explored when training our model. Relu performed poorly from the start and further training was abandoned quickly. Sigmoid and Linear activation were both explored in detail, with Sigmoid narrowly out performing Linear.

4.5 Heatmap Gaussian Sigma 1 vs 4

Initial iterations of the model were trained on ground truth heatmaps which were generated with a Gaussian sigma value of 1. This was the value used commonly in other HPE models, to produce ideal heatmap clouds of 7x7. However, through testing, it was discovered that the heatmaps only contained a single activated pixel. This mistake was due to the downscaling used to reduce heatmap resolution from 256x256 to 64x64 in the data pipeline prior to feeding the heatmaps to the model. The heatmap sigma was adjusted to 1 * 256/64 = 4 to account for the downscaling. An example of the heatmaps before and after the change is shown in Fig. 18

4.6 Testing on Non-Human Photos

To see how the model generalizes to obscure examples, the poses of non-human photos were predicted. These examples can be seen in Fig. 19 and they include cartoons and animals. While these examples do not reflect the intended application of the model, the results were nonetheless interesting.

4.7 Varying Optimizer

We investigated the use of the Adam and rmsProp optimizers and discovered that, even though the paper we referenced for our model used rmsProp, Adam optimizer resulted in a more stable training and validation loss curve, as seen in Fig. 20. Adam strikes a balance between rmsProp and a momentum optimizer, and thus may be balancing the search for a smaller training loss with any regressions from the model on the validation set. Thus, the final models were trained entirely with Adam optimizers.

4.8 Final Parameters

After our experiments, the final parameters we used to develop the most accurate model was 4 hourglass layers (8 was too slow to train), medium augmentation, learning rate scheduler from 5e-3 to 5e-4, Adam optimizer, and Sigmoid activation.

5 Discussion

The experiments we ran aided our model's predictions through one form or another. Data augmentation increased our models ability to generalize. We suspected our model may overfit to a certain orientation so we implemented a left and right flip. We applied left and right flips on input images, then took the average joint locations between the original images and the flipped images. In Fig. 8 and Fig. 9, we can see that there is a significant improvement of the OKS matrix score with the left-right-flip trick applied. We predict that this is due to the bias in the COCO dataset. If the training dataset contains more people facing in a particular direction (left or right), then our model would be more comfortable with test images containing people facing in that direction. Another advantage of applying the left-right-flip trick is that it helps reduce the noise in the original images.

We were surprised to see Relu performing poorly as it usually offers a solution to a common problem of deep networks, vanishing gradient. Nonetheless, both Linear and Sigmoid activations looked promising and were trained further. Both activations performed competitively but Sigmoid provided the best results with an average PCK of 0.7872, a more thorough breakdown of our model's performance using the Sigmoid activation was reported in the evaluation metrics section.

Due to the limitation of time and resources, we were unable to fully train our hourglass model architecture with more than 4 layers. Although with the current data we have, we can confidently predict that an hourglass model with 8 layers will lead to a higher accuracy. Further research should take into account the relationship between the number of hourglass model layers and the accuracy scores.

6 Conclusion

Our final results have shown that the stacked hourglass network performs very well in determining human poses for most images, especially images with well-centered human figures. Compared to other relevant studies, we emphasized on training with a larger dataset (COCO vs MPII), with improved data filtering and random data augmentation. These steps are found useful in model generalizations. Furthermore, we have demonstrated the effectiveness of intermediate supervision, which ensures gradients propagate deep into the network. Together with tuned hyperparameters, our final model achieves a loose OKS metric score of 0.795 and an average PCK score of 0.787, which is competitive with models on the COCO leaderboard from 2016. Averaging predictions from the original image and a horizontally flipped version of the image resulted in a 3-5 % increase in OKS, and a 1% increase in PCK scores.

7 Future Work

Our model currently focuses on images containing a single person. A logical next step would be adding a person detection module to perform top-down multi-person pose estimation. A secondary dataset could be used to validate the model's generalization. A callback to evaluate PCK or OKS after training each epoch could be implemented for quicker model evaluations. To improve model robustness to occluded keypoints, we could randomly remove a large block of the image, while preserving the keypoint ground truth label. This may help the model learn to predict nearby keypoints that are not visible. Our model predictions may be more accurate if we could take a center of mass of the island of activation which contains the most intense activation pixel. This would weigh the local region predictions from the network rather than rely on the network outputting an exact pixel as the most intense pixel, and may improve results. Finally, expanding the training set to include examples without humans could potentially increase the models confidence when predicting 0 keypoints.

References

[1] DeepPose: Human Pose Estimation via Deep Neural Networks. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2014/papers/Toshev_DeepPose_Human_Pose_2014_CVPR_paper.pdf

[2] Efficient Object Localization Using Convolutional Networks. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2015/papers/Tompson_Efficient_Object_Localization_2015_CVPR_paper.pdf

[3] Human Pose Estimation with Iterative Error Feedback. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2016/papers/Carreira_Human_Pose_Estimation_CVPR_2016_paper.pdf

[4] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," 2016.

[5] Y. Chen, Y. Tian, and M. He, "Monocular human pose estimation: A survey of deep learning-based methods," *Computer Vision and Image Understanding*, vol. 192, p. 102897, Mar 2020. [Online]. Available: http://dx.doi.org/10.1016/j.cviu.2019.102897

[6] Common Objects in Context. [Online]. Available: https://cocodataset.org/#home

[7] DensePose. [Online]. Available: http://densepose.org/

[8] MPII Human Pose Database. [Online]. Available: http://human-pose.mpi-inf.mpg.de/#dataset

[9] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[10] S. C. Babu, "A 2019 guide to human pose estimation with deep learning," *Nanonets*, 2019. [Online]. Available: https://nanonets.com/blog/human-pose-estimation-2d-guide/

[11] Common Objects in Context. [Online]. Available: https://cocodataset.org/#keypoints-eval

[12] Common Objects in Context. [Online]. Available: https://cocodataset.org/#format-results

[13] cbsudux/Human-Pose-Estimation-101 | Github. [Online]. Available: https://github.com/cbsudux/Human-Pose-Estimation-101

8 Appendix

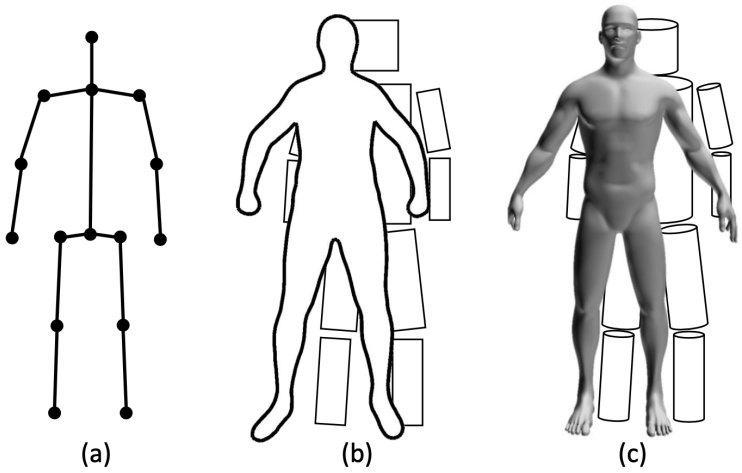


Figure 7: Common body models: (a) skeleton-based, (b) contour-based, (c) volume-based [5]

Table 3: A summary of 2D single-person human pose estimation methods [5]

Methods	Backbone	Input size	Highlights	PCKh (%)
Regression-based				
(Toshev and Szegedy, 2014)	AlexNet	220×220	Direct regression, multi-stage refinement	-
(Carreira et al., 2016)	GoogleNet	224×224	Iterative error feedback refinement from initial pose.	81.3
(Sun et al., 2017)	ResNet-50	224×224	Bone based representation as additional constraint, general for both 2D/3D HPE	86.4
(Luvizon et al., 2017)	Inception-v4+ Hourglass	256×256	Multi-stage architecture, proposed soft-argmax function to convert heatmaps into joint locations	91.2
Detection-based				
(Tompson et al., 2014)	AlexNet	320×240	Heatmap representation, multi-scale input, MRF-like Spatial-Model	79.6
(Yang et al., 2016)	VGG	112×112	Jointly learning DCNNs with deformable mixture of parts models	-
(Newell et al., 2016)	Hourglass	256×256	Proposed stacked Hourglass architecture with intermediate supervision.	90.9
(Wei et al., 2016)	CPM	368×368	Proposed Convolutional Pose Machines (CPM) with intermediate input and supervision, learn spatial correlations among body parts	88.5
(Chu et al., 2017)	Hourglass	256×256	Multi-resolution attention maps from multi-scale features, proposed micro hourglass residual units to increase the receptive field	91.5
(Yang et al., 2017)	Hourglass	256×256	Proposed Pyramid Residual Module (PRM) learns filters for input features with different resolutions	92.0
(Chen et al., 2017)	conv-deconv	256×256	GAN, stacked conv-deconv architecture, multi-task for pose and occlusion, two discriminators for distinguishing whether the pose is 'real' and the confidence is strong	91.9
(Peng et al., 2018)	Hourglass	256×256	GAN, proposed augmentation network to generate data augmentations without looking for more data	91.5
(Ke et al., 2018)	Hourglass	256×256	Improved Hourglass network with multi-scale intermediate supervision, multi-scale feature combination, structure-aware loss and data augmentation of joints masking	92.1
(Tang et al., 2018a)	Hourglass	256×256	Compositional model, hierarchical representation of body parts for intermediate supervision	92.3
(Sun et al., 2019)	HRNet	256×256	high-resolution representations of features across the whole network, multi-scale fusion.	92.3
(Tang and Wu, 2019)	Hourglass	256×256	data-driven joint grouping, proposed part-based branching network (PBN) to learn representations specific to each part group.	92.7

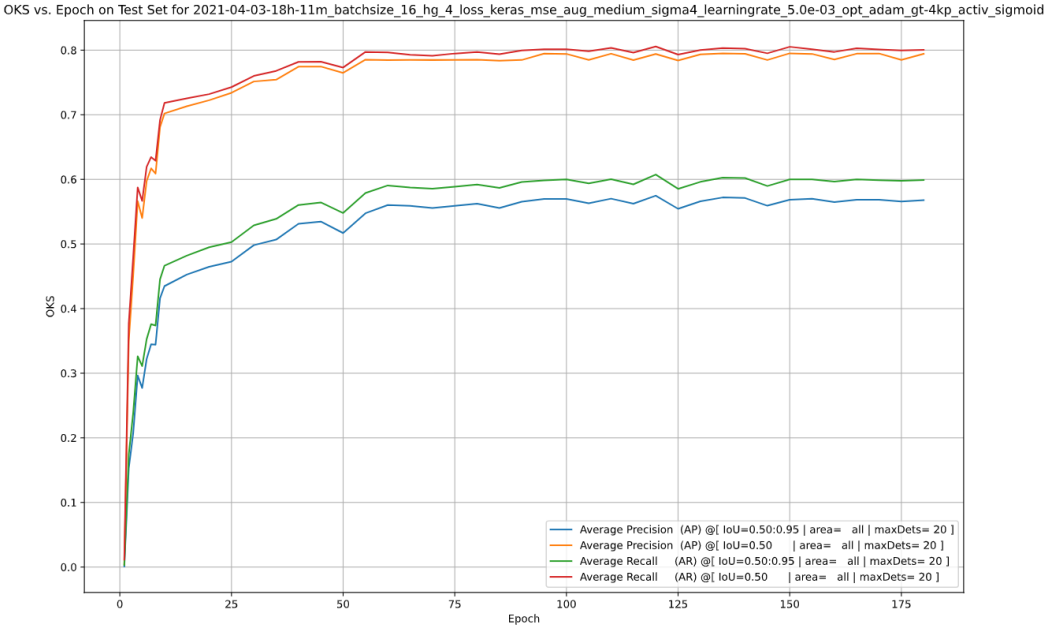


Figure 8: COCO Evaluation OKS Metric Across Epochs for Hourglass 4, Averaging Normal and L/R flip predictions

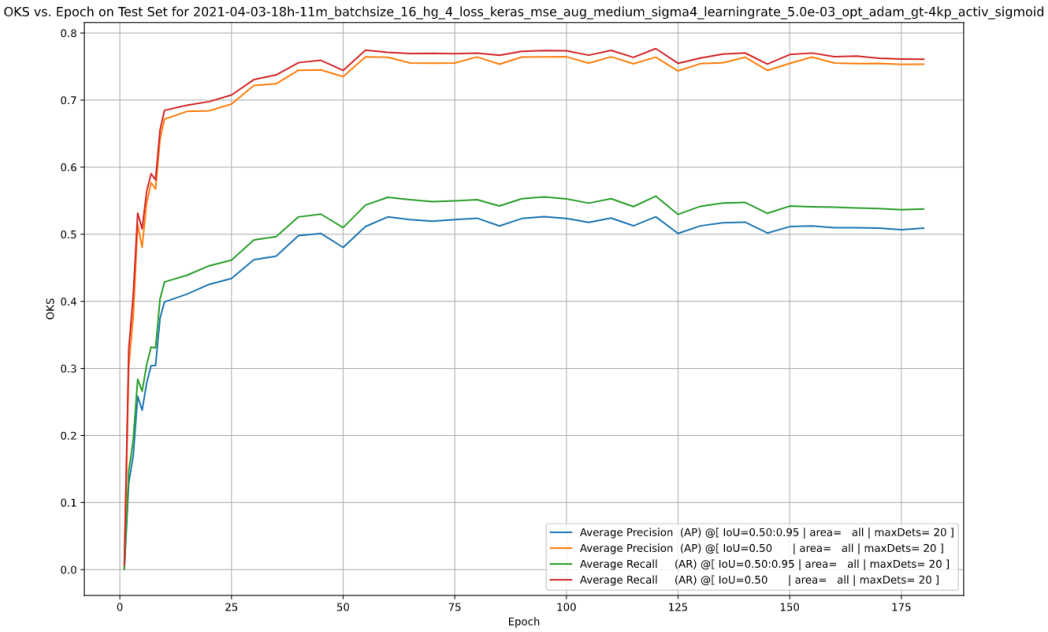


Figure 9: COCO Evaluation OKS Metric Across Epochs for Hourglass 4, No Averaging Normal or L/R flip predictions

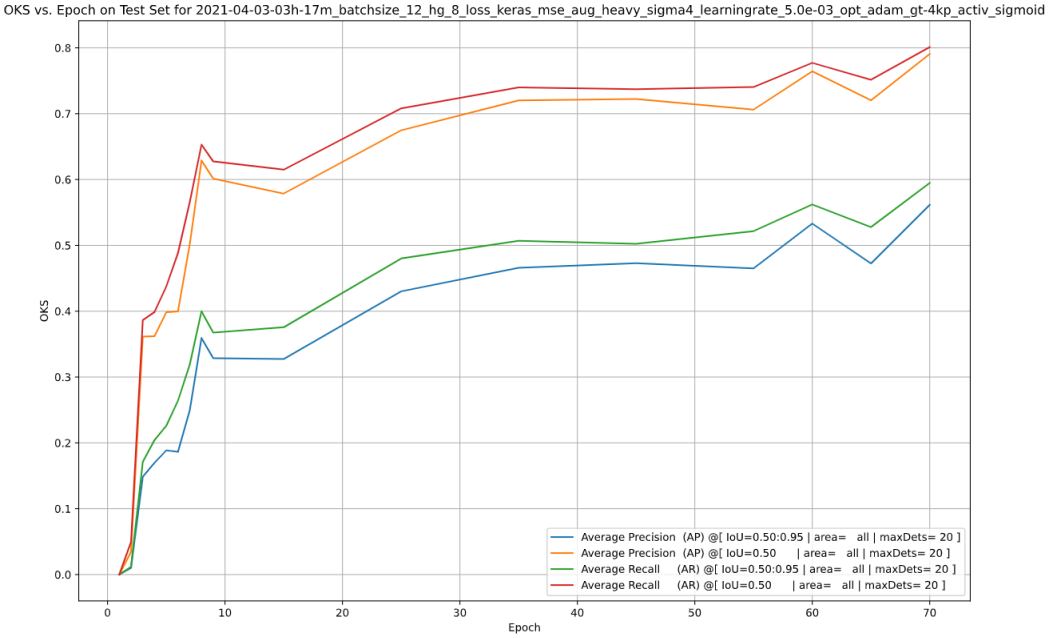


Figure 10: 8-layer Hourglass Model OKS Matrix Score

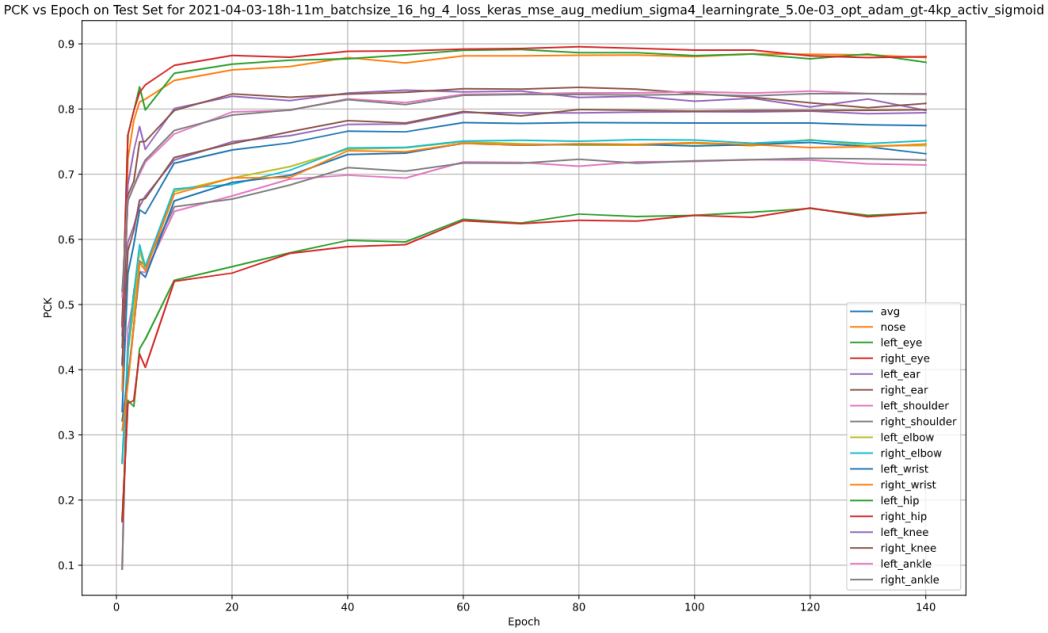


Figure 11: COCO Evaluation PCK Metric Across Epochs for Hourglass 4, No averaging of normal and L/R flip predictions

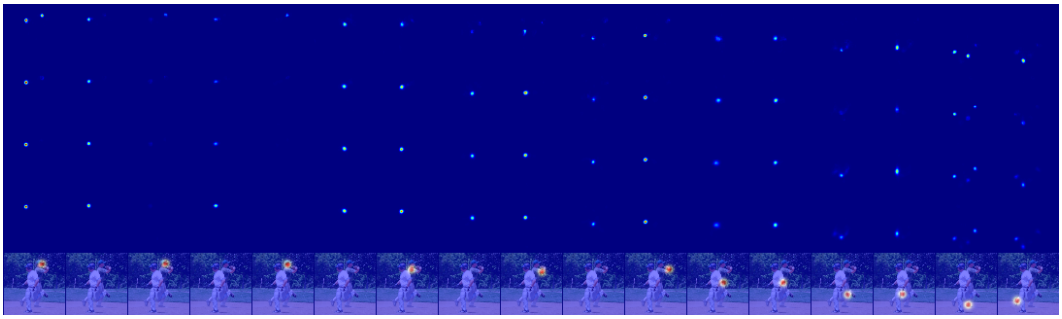


Figure 12: 4-layer hourglass heatmaps of overlaid baseball players (Back)

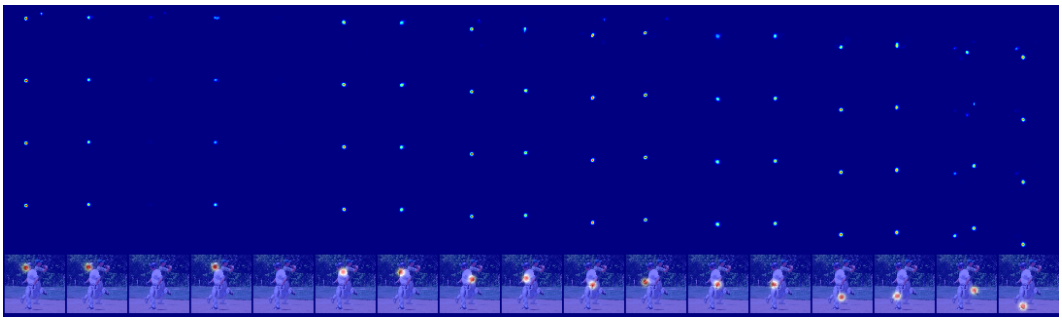


Figure 13: 4-layer hourglass heatmaps of overlaid baseball players (Front)

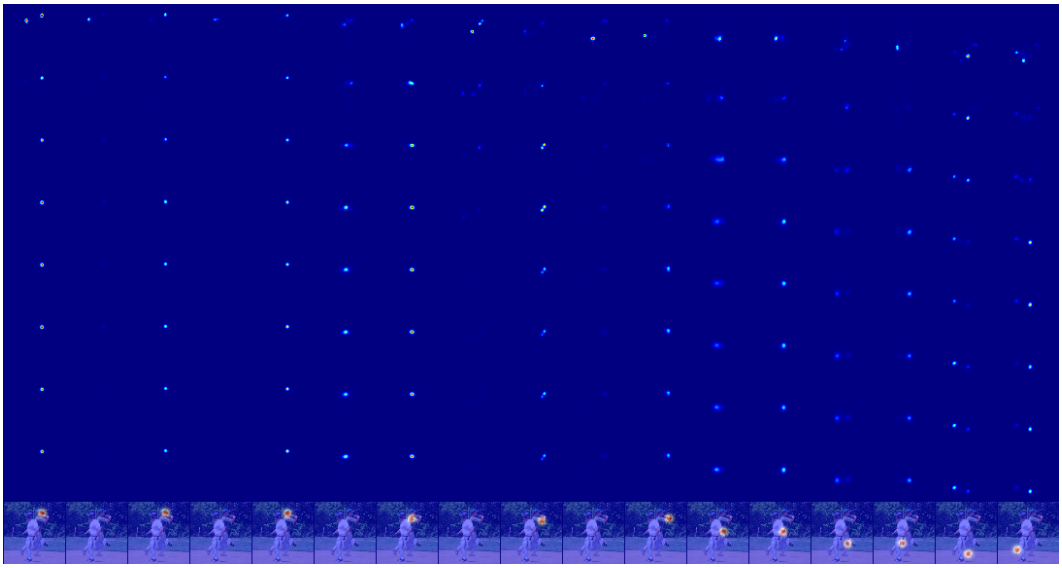


Figure 14: 8-layer hourglass heatmaps of overlaid baseball players (Back)

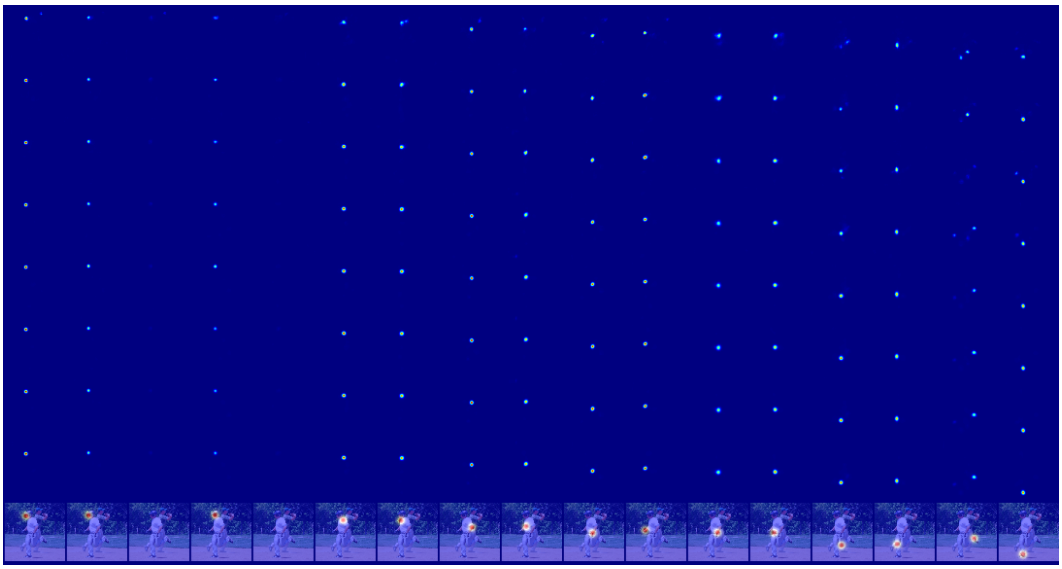


Figure 15: 8-layer hourglass heatmaps of overlaid baseball players (Front)

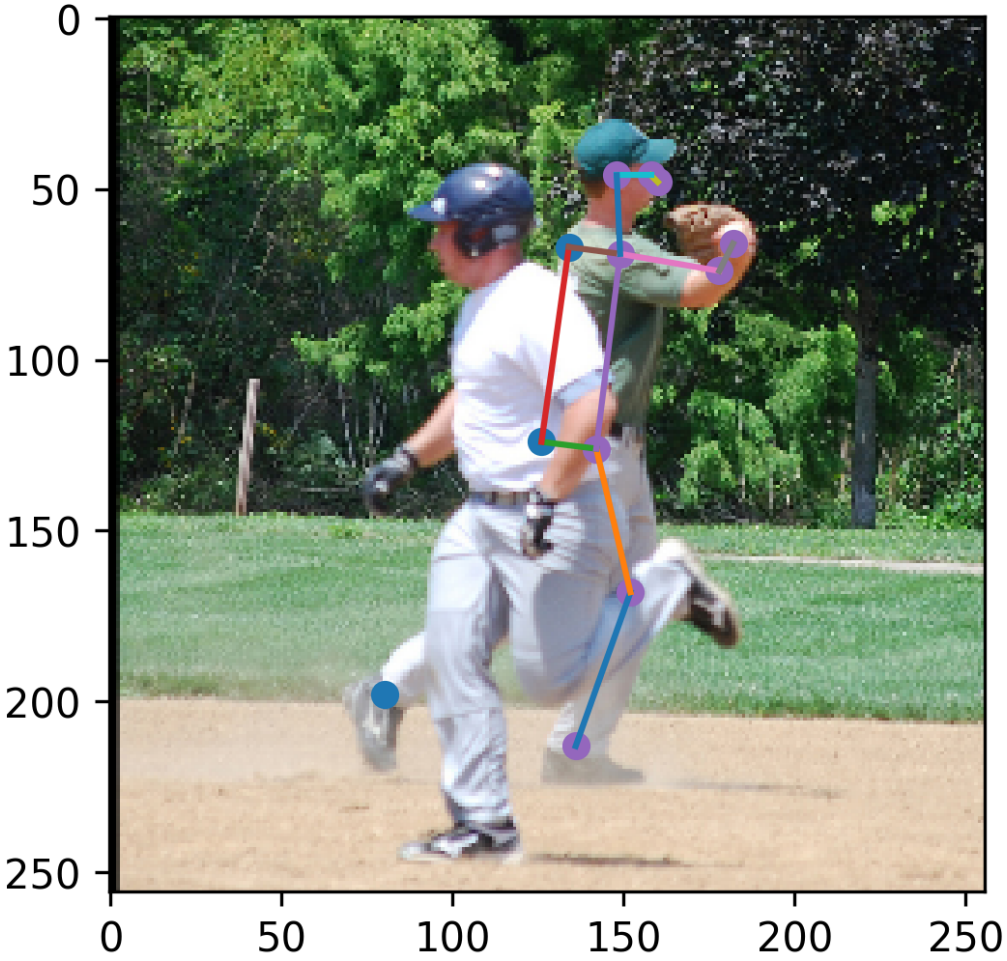


Figure 16: Keypoint skeleton correctly drawn on background player

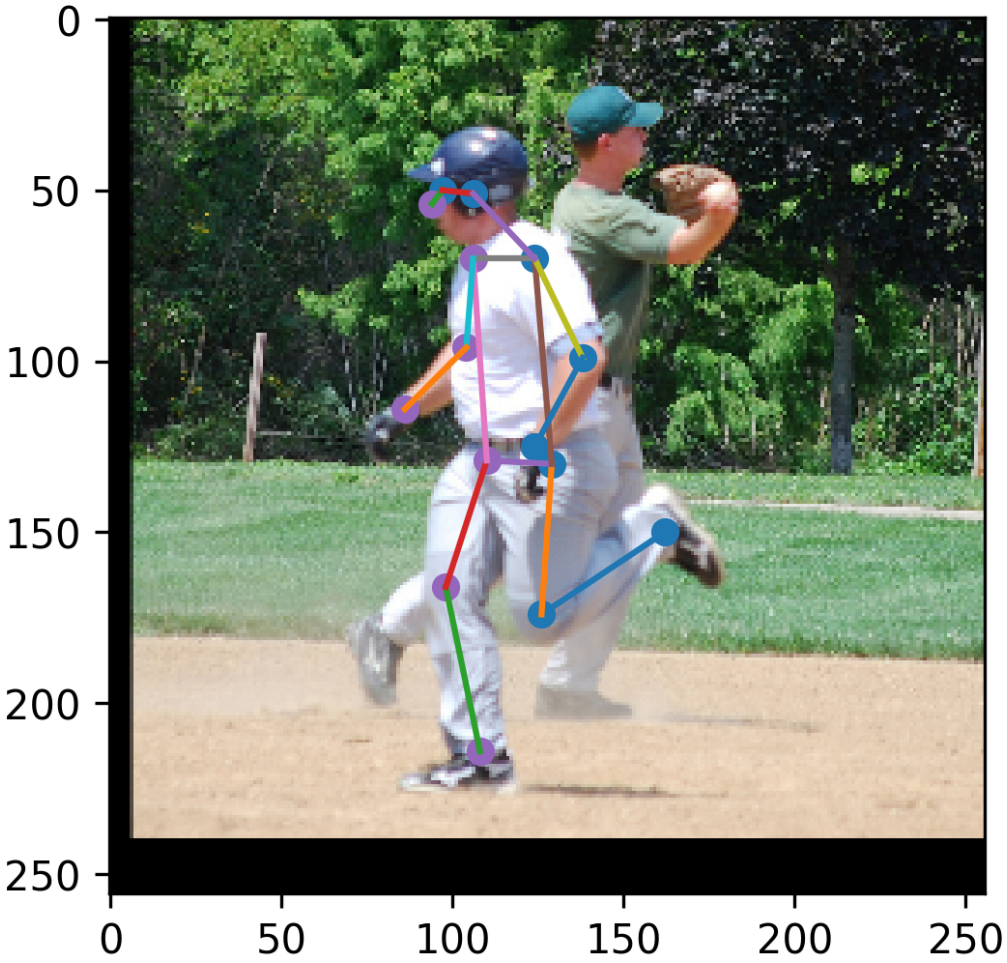
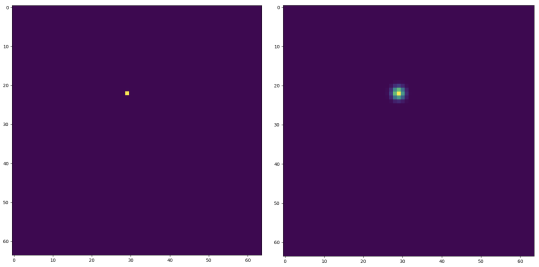
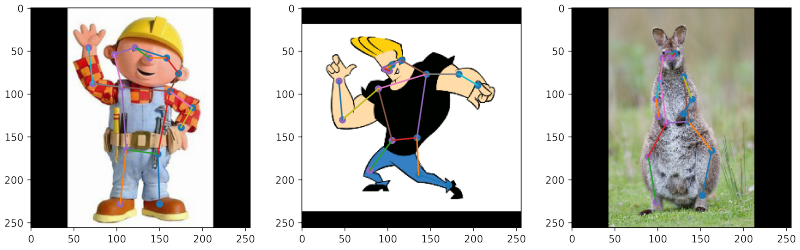


Figure 17: Keypoint skeleton correctly drawn on foreground player



(a) Sigma = 1 (b) Sigma = 4

Figure 18: Example of heatmaps generated with different Gaussian Sigma



(a) Found all keypoints except knees

(b) Found all keypoints except feet

(c) Found keypoints by observing general patterns

Figure 19: Non human photos

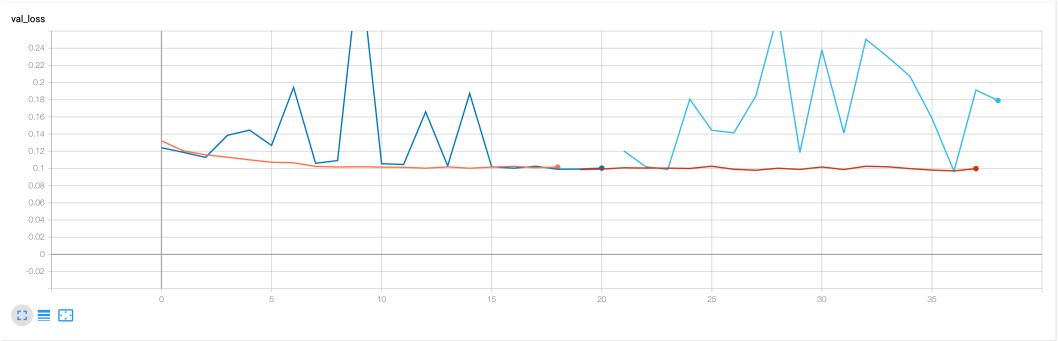


Figure 20: Validation loss, Adam optimizer (orange) vs. rmsProp (blue)

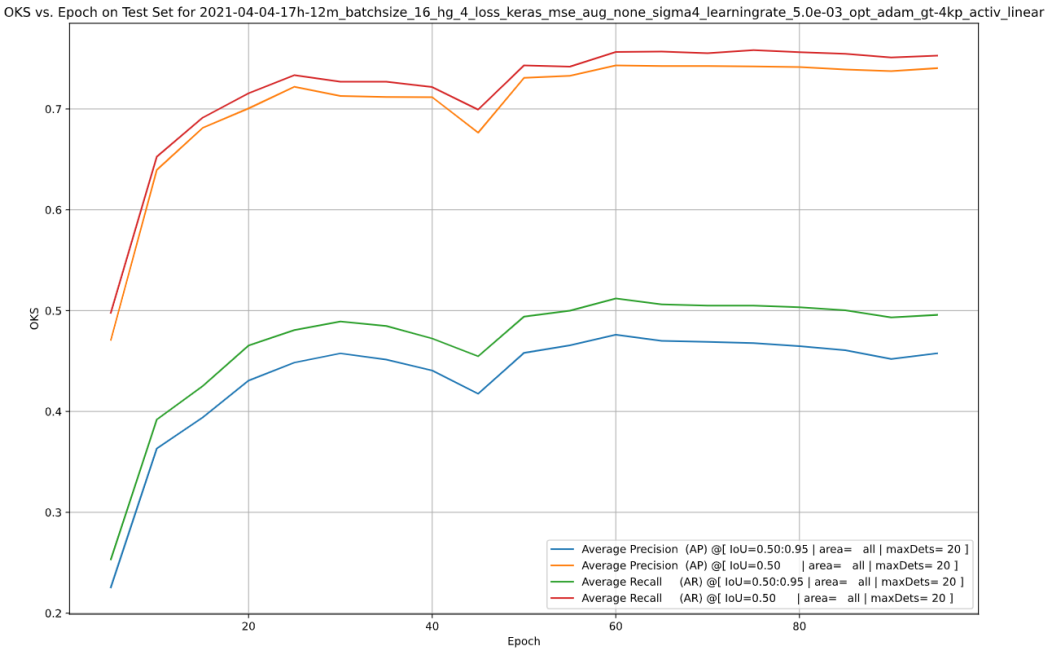


Figure 21: No augmentation, showing a max score approximately 5% below