2D Alpine Skier Pose Estimation

Longling Tian
ICME
Stanford University
longling@stanford.edu

Xingshuo Xiao
ICME
Stanford University
xingshuo@stanford.edu

Abstract

With the advancements in deep learning and computer vision models, it is possible to produce an accurate, automatic, and ideally real-time skier pose estimation. In this project, we investigate the performances of different CNN models for 2D alpine skier pose estimation on the Ski 2DPose Dataset both quantitatively and qualitatively. Our findings suggest that the keypoint R-CNN model shows promising results in estimating the poses of alpine skiers across metrics like Mean Per Joint Position Error (MPJPE), Percentage of Keypoints (PCK), and Percentage of Correct Parts (PCP), compared to VGG19 and ResNet-50. Through our experiments, we demonstrate the potential of enhancing pose estimation accuracy for alpine skiing and provide insights for further research in this field, such as additional training on distant view photos.

1. Introduction

Alpine skiing is a thrilling winter sport involving high speed and sweeping turns from steep slopes. It is highly demanding in movement control and flexibility. An accurate pose estimation for alpine skiing would play an important role in coaching and judging. Coaches could analyze and evaluate the skiers' performances through real-time pose estimation in order to find an optimal movement trajectory and prevent skiers from injuries.

The existing techniques for skier pose analysis include the setup of body-worn sensors [22] [8], which is expensive and time-consuming. Alternatively, the coaches and athletes could investigate their performances through the videos. However, it is tedious and highly subjective to manually annotated through the videos. Therefore, an accurate, automatic, and ideally real-time skier pose estimation model would be crucial.

With the advancements in deep learning and computer vision techniques, pose estimation has emerged as a promising approach to automatically infer the skeletal structure of humans and animals. The deep-learning-based human pose estimation methods can generate relatively solid predictions for both 2D and 3D poses. The dynamic nature of sports activities distinguishes sports pose estimation from general human pose estimation. The sports activities involve high-speed motion, more variances in poses, and diverse environment factors, which together contribute more challenges to pose estimation. In addition, different sports emphasize different key joints. For example, in gymnastics, the position of the hips and knees may play a more critical role than they do in basketball. Understanding the sport-specific key joints is essential for effective pose estimation. Thus, sports-specific pose estimation algorithms are needed to enhance accuracy and robustness catering specifically to the demands of sports activities.

To examine the performance of 2D pose estimation algorithms for alpine skiers, we tested a keypoint R-CNN model on Ski 2DPose Dataset [1]. Through quantitative and qualitative evaluations of the keypoints detection, our goal is to find potential approaches to enhance the pose estimation accuracy.

2. Related Work

2.1. Deep-learning-based Pose Estimation

Human pose estimation has been a popular research topic since the rise of deep learning. Numerous methods have been proposed in literature for both 2D and 3D pose estimation. For 2D human pose estimation specifically, it can be categorized into single-person and multi-person pose estimation depending on the objects of interests. According to Dang et al. [6], there are two approaches, direct regression-based approaches and heatmap-based approaches, to solve single-person pose estimation problems. Direct regression-based approaches predict the keypoint locations based on the features directly, while heatmap-based approaches first generate a heatmap indicating the probability that a keypoint exists in the particular region and make prediction with the heatmap. For multi-person poses, the approaches could be classified into either top-down or bottom-up approaches. The top-down approaches first de-

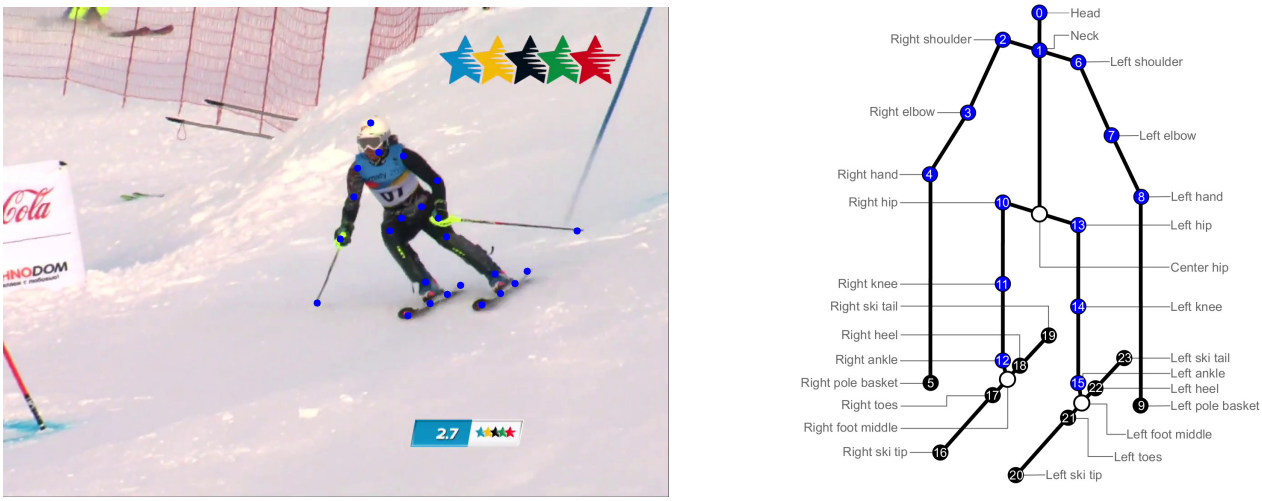

Figure 1. An example of annotated images and the annotation diagram.

tect the humans from the input image and then predict the keypoints, while the bottom-up approaches work reversely: they find the keypoints first and then group the keypoints that belong to the same person.

There are a number of 2D human pose estimation models proposed. Some notable examples include OpenPose [4], HRNet [21] [5], and AlphaPose [7]. Cao et al. first proposed OpenPose as a real-time pose estimation model following a bottom-up framework to detect in total 135 keypoints of people in an image. AlphaPose enables simultaneous human pose tracking within a top-down framework. Sun et al. presents HRNet that maintains high-resolution representations through the entire network. HigherHRNet was built on top the highest resolution feature maps of HRNet to generate a more accurate predictions with additional deconvolution modules. Although these algorithms have achieved promising accuracy, the efficiency could be further improved by simplifying the models.

2.2. Skier Pose Estimation

Even though there are many datasets and models proposed for sports pose estimation [17] [24] [25] [13], there are very limited number of prior work investigating pose estimation for skiing due to the lack of large-scaled datasets that cover all disciplines. Some researchers have worked on 3D pose estimation for ski jumpers [26] [2], but ski jumping does not involve turns, hence the models might not be able to accurately predict the keypoints for alpine skiers. Rhodin et al. [19] utilizes multi-view images for 3D alpine skier pose estimation tasks. It works effectively with a small-scale dataset. Bachmann et al. [1] integrate rotation estimation of pan-tilt cameras to optimize feature tracking for the high speed and background variations, and ultimately generate 3D poses. 3D skier pose estimation could provide more insights on the skiers' motions in the space, however, it requires more depth sensors and multiple calibrated cameras which might be challenging.

2.3. R-CNN Model

One of the biggest challenges in object detection is that there are too many regions to search for. Girshick et al. proposed region-based convolutional neural networks (R-CNN) [10] as a solution to the problem. Through selective search, the algorithm extracts 2000 candidate regions. Then, the classifier recursively combines similar regions into larger ones to generate final region proposals. The same group later proposed fast R-CNN [9] to speed up the algorithm by extracting the features from the entire image first using a CNN. Ren et al. introduces faster R-CNN [18] which further lowers the running time by avoiding selective search in R-CNN and fast R-CNN.

Mask R-CNN [11] extends faster R-CNN with an additional object mask prediction with the existing object bounding boxes. Mask R-CNN is able to solve more problems such as instance segmentation.

Using a keypoint detection instead of mask prediction in mask R-CNN, keypoint R-CNN [23] allows us to make human pose estimation.

3. Data

3.1. Dataset

For this project, we used Ski 2DPose Dataset [1]. This dataset consists of 16 alpine skiing Youtube videos from different perspectives in various weather conditions and backgrounds. The original dataloader has videos split into 147 training and 11 validation video sequences, while each representing a continuous camera motion without any cuts.

The dataset consists of 1982 images of at least 32 amateur to semi-professional alpine skiers, with 17 being women and 15 being men. The dataset includes footage from 5 distinct locations, captured under various weather conditions ranging from sunny to foggy. The images are either 1920 $\times$ 1080 or 1280 $\times$ 720 large with a resolution of 96 dpi.

Within these images, 24 joints, including skis and poles, were manually annotated as shown in Figure 1. The labels and image information are stored in a JSON file. The keypoint annotations are organized in the form of (x, y, visibility). The coordinates (x,y) range from 0 to 1, where (0,0) represents the upper-left corner and (1,1) represents the lower-right corner. The visibility flag is 0 when the joint is invisible in the image, and it is set to 1 when it is visible.

3.2. Training/Validation/Test Data Split

The Ski 2DPose Dataset does not have test data provided. Therefore, we rearranged the training and validation splits by a rough ratio of 8:1:1. The detailed distribution is presented in Table 1.

	# of images	# of video sequences
Training	1602	132
Validation	190	16
Test	190	10

Table 1. Training/validation/test data split distribution.

3.3. Pre-processing

We adapted the dataloader class provided by Bachmann et.al. [1] and made necessary modifications to fit the different models.

For the baseline models, the dataloader outputs a normalized image tensor with the size of $[N, 3, 224, 224]$ where N is the batch size, and an annotation tensor of 24 $\times$ 2 (i.e. x-y coordinate of the 24 joints). We mapped x coordinates from $[0, 1]$ to $[0, W]$, and y from $[0, 1]$ to $[0, H]$, where W and H are the image width and height after resizing.

For the keypoint R-CNN model, the dataloader outputs an image tensor and a target dictionary. The image tensor has a size of $[N, 3, 1080, 1920]$ with all original images normalized. For the target dictionary has the following keys and values format:

- **boxes** - A tensor of size $[N, 4]$, where $N = 1$ indicates the number of detected objects (the skier) in the image. Each box is arranged in the form $[x_1, y_1, x_2, y_2]$.
- **labels** - A tensor of size $[N]$, indicating the class of the object. For our project, it is always 1 as we only have one skier in a image. 0 represents the background.

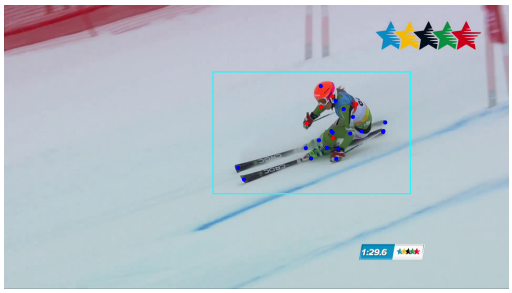

Figure 2. Example of bounding box and joint annotation. The blue points are visible joints and the red points are invisible ones.

- **keypoints** - A tensor of size $[N, 24, 3]$ depicting the 24 joint positions of the skier. Each joint position is in the form (x, y, flag).
- **area** - A tensor of size $[N]$ indicating the area of the bounding box.
- **iscrowd** - A tensor of size $[N]$. If `iscrowd = true`, the object would be ignored during evaluation.

Because the dataset only gives annotated ground truth, in order to fit the model, we annotated the box by finding the minimum and maximum xy coordinates from the keypoints. The resulting bounding box is $(\min(x_{min} - 0.05, 0), \min(y_{min} - 0.05, 0), \max(x_{max} + 0.05, 1), \max(y_{max} + 0.05, 1))$. The annotation for keypoint R-CNN model is shown in Figure 2.

4. Methods

In this project, we performed the pose estimation on a baseline model and a keypoint R-CNN model with three different evaluation metrics discussed as follows.

4.1. Baseline Model

The baseline models we used are pretrained convolutional neural network built on PyTorch [2]. We used transfer learning with ResNet-50 and VGG19. The models are pretrained on ImageNet and retrained with a finetuned output layer. When we worked on the baseline models, we only used the (x, y) coordinates for the keypoints information and ignored the visibility flag due to the limited number of training images and the nature of evaluation metrics.

The ResNet-50 model is a CNN model with 50 layers [12], while the VGG19 model has 19 layers [20]. Both models take an input image with a size of 3 $\times$ 224 $\times$ 224, and would output a vector with a length of 48 for both x and y coordinates in the form $(x_1 \quad y_1 \quad x_2 \quad y_2 \quad ...)$.

[1] https://learnopencv.com/human-pose-estimation-using-keypoint-rcnn-in-pytorch/#overview

[2] https://github.com/Cadene/pretrained-models.pytorch#torchvision

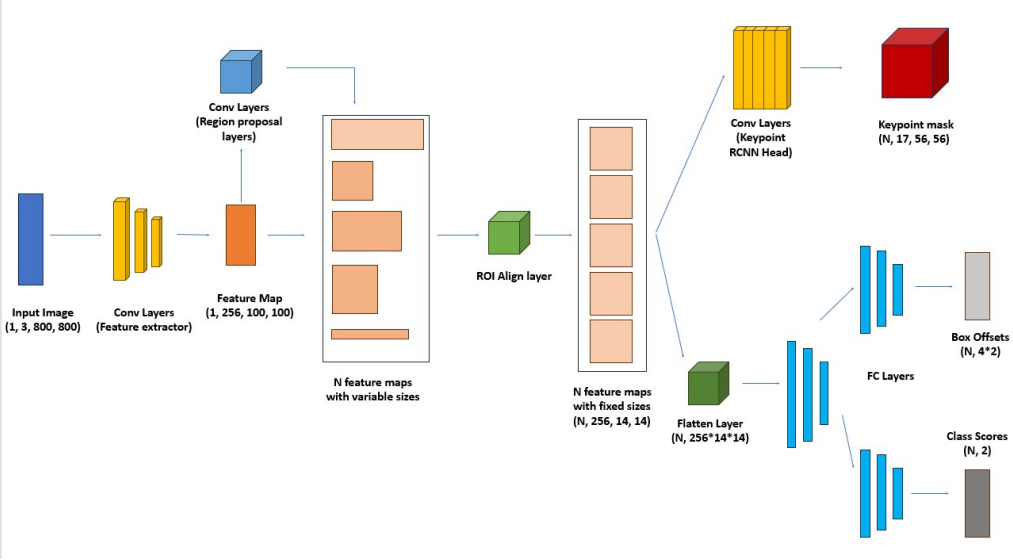

Figure 3. Pretrained Keypoint R-CNN model architecture[1]

4.2. Keypoint R-CNN

Keypoint R-CNN model is a pretrained model written in PyTorch TorchVision library based on Mask R-CNN [11, 16]. The model is pretrained on MS-COCO dataset which would detect 17 keypoints [15]. This is a one-stage pose estimation method which combines person detection and pose estimation. Compared to the two-stage strategies, either top-down or bottom-up approach, the one-stage technique has a higher efficiency [14].

The architecture of Keypoint R-CNN is similar to the Mask R-CNN with a different output size for keypoint mask (Figure 3). Instead of encoding the whole object mask in Mask R-CNN, Keypoint R-CNN highlights the locations for the joints in the keypoint mask.

This model is built with the ResNet-50 FPN backbone to generate the primary feature maps. At the training mode, the model returns a dictionary containing regression and classification losses for the region proposal network (RPN) and R-CNN, in addition to the keypoint losses.

At the evaluation mode, the model outputs a dictionary with estimation for bounding boxes and keypoints with corresponding confidence scores for each box and keypoint in the following format:

- **boxes** - A tensor of size $[N, 4]$ with the predicted boxes in $[x_1, y_1, x_2, y_2]$ format.
- **labels** - A tensor of size $[N]$ with the predicted label for each corresponding predicted box.
- **scores** - A tensor of size $[N]$ with the confidence scores for each predicted box.
- **keypoints** - A tensor of size $[N, 24, 3]$ with the coordinates of predicted keypoints.
- **keypoints_scores** - A tensor of size $[N, 24]$ with the confidence scores for each predicted keypoint.

However, the pretrained Keypoint RCNN model available in Pytorch TorchVision library [11] was not applicable to this study, because the pretrained model has 17 keypoints, whereas we have 24. Therefore, we were not able to apply any transfer learning and had to retrain all parameters from scratch according to the tutorial[3]. We retrained the weights while kept the pretrained ResNet-50 FPN backbone.

4.3. Evaluation Metrics

We used four evaluation metrics for this keypoint detection task. Here k_{pred} and k_{true} are 24×2 matrices representing predicted and ground truth keypoint pixel values.

$$\text{MPJPE} = \frac{1}{N_k} \sum_{i=1}^{N_k} ||k_{pred,i} - k_{true,i}||_2$$

The first one, namely Mean Per Joint Position Error (MPJPE), measures the average Euclidean distance between predicted keypoint and ground truth, across all $N_k = 24$ joints.

The second and third metric are based on percentage of correctly identified keypoints. The difference is how we identify a prediction to be correct.

[3]https://medium.com/@alexppppp/how-to-train-a-custom-keypoint-detection-model-with-pytorch-d9af90e111da

$$\text{PCK}_5 = \frac{1}{N_k}\sum_{i=1}^{N_k}\delta(||k_{pred,i} - k_{true,i}||_2 \leq 5)$$

Percentage of Correct Keypoints (PCK) classifies a prediction to be accurate if it's within a certain threshold of pixel values from the ground truth label. In this study, we arbitrarily chose this value to be 5 pixels. This is based on the average length and width of human on the images.

$$\text{PCP} = \frac{1}{N_b}\sum_{j=1}^{N_b}\delta(||k_{pred,B_{j,1}} - k_{true,B_{j,1}}||_2 \leq L_j/2) \cdot \delta(||k_{pred,B_{j,2}} - k_{true,B_{j,2}}||_2 \leq L_j/2)$$

And on the other hand, one can choose the thereshold based on the attributes of each individual image. To be precise, we applied the third evaluation metric called Percentage of Correct Parts (PCP). This metric considers not the keypoints, but the parts (or bones) connecting neighboring keypoints. N_b represents the number of bones (in our dataset, $N_b = 25$), and B is a 25×2 matrix representing the index of start and end point for a bone. For example, $B_7 = [2, 3]$ means the 7th bone starts from 2nd keypoint and ends at 3rd keypoint. And L is a vector specifying bone length. A bone is considered correctly identified if both the start point and end point is within an error from the ground truth half the bone's length. (The ratio 0.5 is widely used among studies on keypoint detection.)

$$\text{PCK}_{\text{torso}} = \frac{1}{N_k}\sum_{i=1}^{N_k}\delta(||k_{pred,i} - k_{true,i}||_2 \leq t)$$

Similarly, we can modify the PCK metric based on individual images. In practice, we chose the thereshold to be within 0.2 times torso diameter t, which is the distance from neck to hip. The difference between this and PCP is that PCP has different theresholds for each bone, while $\text{PCK}_{\text{torso}}$ has uniform threshold for all joints within an image.

For this study, we did not use average recall (AR) or average precision (AP) as evaluation metrics. The main reason comes from the calculation of Object Keypoint Similarity (OKS). In order to calculate AR or AP values, one needs to calculate average values over different OKS thresholds. However, the per-keypoint constant in OKS was not specified on the dataset. Therefore, we carried out this study using the three evaluation metrics listed above.

5. Experiments

The experiments were conducted using an NVIDIA T4 GPU unit.

During the baseline model training, the batch size was set to 15. We used an Adam optimizer ($\beta_1 = 0.9, \beta_2 = 0.999$) with a weight decay of $1e-7$ on the model parameters. MSE loss is backpropagated at each batch step, and the model undergoes training for 20 epochs. To have a faster convergence and avoid overfitting, we applied a StepLR scheduler with an initial learning rate of $1e-3$.

For the R-CNN model, We used SGD optimizer over Adam because Adam sometimes failed to converge on the training set. SGD, on the other hand, had overall smaller training loss but took more epochs to converge. Therefore we set number of epochs to 25 and applied StepLR learning rate scheduler and early stopping based on the behavior on validation set. The initial learning rate was set to $1e-3$.

We chose a batch size of 10 for the RCNN model because of the limited GPU memory. A larger batch size (e.g. $\geq$15) would lead to CUDA memory error, while smaller batch size took longer time to train.

5.1. Results

	MPJPE	PCK_5	PCP	$\text{PCK}_{\text{torso}}$
VGG19	16.5	12.7%	12.7%	13.3%
ResNet-50	13.7	17.6%	12.6%	13.1%
Keypoint R-CNN	8.3	55.4%	29.2%	31.6%

Table 2. Model Performance on Test Set

Table 2 demonstrates the performance of baseline ResNet-50, baseline VGG19 and Keypoint R-CNN model on the test set. MPJPE is measured in pixel values, and the other three are in percentages. We can see that Keypoint R-CNN achieved better performance than VGG19 and ResNet-50 in all 4 metrics.

Compared to the baseline ResNet-50, our Keypoint R-CNN model generates the keypoint estimation with only the features from the selected regions. The ROI align layer and keypoint R-CNN head predict the mask for each keypoint [3]. Thus, instead of considering the features from the entire image, the R-CNN model is more efficient and could take fewer noises in prediction.

Moreover, in the baseline models training process, we only considered the (x, y) coordinates of the keypoints. Thus, when the keypoints are occuluded, the R-CNN model would have a greater chance to make the accurate prediction.

One may notice that the PCK_5 value was especially high compared to other metrics. This is because in PCK_5, the value of 5 pixels was a hard thereshold, compared to PCP and $\text{PCK}_{\text{torso}}$, which are relative to human body size in pixels. Since the images in our datasets are all extracted from video frames, it's inevitable that the size of human body are not uniform across all images. Specifically, there are certain

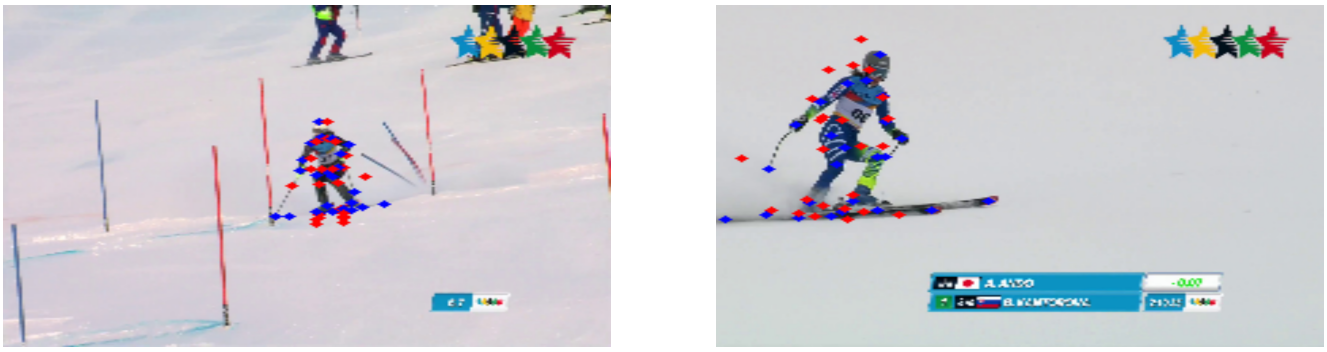

Figure 4. An example of generally correct prediction (left) and inaccurate prediction (right).

images where human body only takes less than 100 pixel values. In this case a deviation of 5 pixels would be very easy to achieve, thus the PCK_5 value being high. Therefore in practice, we would care more about PCP and PCK_{torso}.

6. Discussion

By visualizing the predicted keypoints, we found that the qualitative performances of the models are very inconsistent for different keypoints. For example, Figure 4 shows two examples of the Keypoint R-CNN model prediction. The blue dots depict the ground truth and the red dots are the predicted keypoints. The left image represents a common success case. However, we can still see that some dots at head, poles, and ski tips are not well-detected. One possible reason is that skiers' upper bodies are often not upright. Also, because of the ski jackets they wear, the colors across the whole body are usually all the same, and some joint features might be covered by the thick fabrics, which make increase the difficulty for keypoint detection.

The camera angles could be a factor contributing to the failure of ski tips detection. We notice that for the failure case in Figure 4, even if the model failed to detect keypoints in the upper body accurately, it did better for the ski tips than the success case. This image was taken from a lower camera angle so that if we zoom in, we could see that the skis clearly. For those images where skis are flat on the ground, snow between two skis may be hidden, and the model may fail to detect both skis from the image.

Furthermore, one interesting trend among failure cases are that all detected keypoints deviate from the body by a certain amount, but the gesture was still kept. This could be illustrated by the red dots in the failure case in image 4. If we move all the red dots to the right by about 20 pixels, they will match the body parts very closely. Besides, during training time, very often the training loss dropped to a certain amount, and then stopped before converging to zero.

We came up with three potential reasons on this. First, we suspect that the optimizer had not converged to minimum. However, even if we trained for 50 epochs the problem still existed. Therefore, the problem may come from the training data and our Keypoint R-CNN model. Since this problem happens more on images where the skiers are farther from the cameras, we might need to have more training data on distant-view photos.

Finally, the bounding box approach in the Keypoint R-CNN model might cause this issue. Since the bounding boxes were generated manually based on the keypoint labels (i.e. no ground truth bounding boxes), they might not be precise enough for the model training. In practice, all bounding boxes are rectangular. But within the box, due to the length of skis and poles, the bodies only take up a small part in the box, while more than half of the box is snow (background). This could explain why this model produces better pose estimation on MS COCO Dataset when the target keypoints only include body parts [14].

7. Conclusion

In this study, we evaluated the performance of VGG19, ResNet-50, and Keypoint R-CNN models for 2D alpine skier pose analysis. Our findings indicate that the Keypoint R-CNN model outperformed the other two models across all four evaluation metrics by 15 to 40 percent. The improved performance of Keypoint R-CNN can be attributed to its efficient feature extraction to detect keypoints, which makes it less vulnerable to background noise in the image.

However, we observed that the Keypoint R-CNN model performed better on close-up images compared to distant view ones. Besides, the model faced challenges in accurately detecting keypoints on the skiers' upper body and ski tips.

In the future, if we have time and more computing resources for this study, we would consider constructing another image segmentation model before the keypoint detection model instead of a box. Segmentation model would clearly capture the shape of human body (in context of skiers' it will be easier because the boundary between dark

ski jacket and white snow will be very sharp). This approach would confine the keypoint detection within the boundaries of the human body, mitigating the issue of detecting keypoints outside the skier's body. Implementing a model with better feature extraction such as Higher HRNet may also improve the results.

Additionally, we can train the model using more distant view pictures. Also, exploring modifications to the backbone structure of the model or incorporating pretrained weights may yield further enhancements. However, implementing these changes would likely require incorporating an additional model to handle the transformation between the 17 keypoints present in the pretrained model and the 24 keypoints used in this study.

8. Contributions & Acknowledgements

We work on this project jointly with a work division as follows:

- Longling Tian: data pre-processing, baseline models evaluation, Keypoint R-CNN model training and evaluation, report writing
- Xingshuo Xiao: literature review, data pre-processing, baseline models training and evaluation, Keypoint R-CNN model training, data visualization, report writing

Our baseline models are obtained from the repository https://github.com/Cadene/pretrained-models.pytorch#torchvision, and the Pytorch Keypoint RCNN model is finetuned on top of the `keypointrcnn_resnet50_fpn` model in PyTorch TorchVision library [16] with reference to https://medium.com/@alexppppp/how-to-train-a-custom-keypoint-detection-model-with-pytorch-d9af90e111da.

References

[1] Roman Bachmann, Jörg Spörri, Pascal Fua, and Helge Rhodin. Motion capture from pan-tilt cameras with unknown orientation. (arXiv:1908.11676), Aug 2019. 1, 2, 3

[2] Wenxia Bao, Tao Niu, Nian Wang, and Xianjun Yang. Pose estimation and motion analysis of ski jumpers based on eca-hrnet. *Scientific Reports*, 13(1):6132, Apr 2023. 2

[3] A-M Boutsi, N Bakalos, and C Ioannidis. Pose estimation through mask-r cnn and vslam in large-scale outdoors augmented reality. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, (4), 2022. 5

[4] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 2

[5] Bowen Cheng, Bin Xiao, Jingdong Wang, Honghui Shi, Thomas S. Huang, and Lei Zhang. Higherhrnet: Scale-aware representation learning for bottom-up human pose estimation. (arXiv:1908.10357), Mar 2020. arXiv:1908.10357 [cs, eess]. 2

[6] Qi Dang, Jianqin Yin, Bin Wang, and Wenqing Zheng. Deep learning based 2d human pose estimation: A survey. *Tsinghua Science and Technology*, 24(6):663–676, 2019. 1

[7] Hao-Shu Fang, Jiefeng Li, Hongyang Tang, Chao Xu, Haoyi Zhu, Yuliang Xiu, Yong-Lu Li, and Cewu Lu. Alphapose: Whole-body regional multi-person pose estimation and tracking in real-time, 2022. 2

[8] Benedikt Fasel, Jörg Spörri, Matthias Gilgien, Geo Boffi, Julien Chardonnens, Erich Müller, and Kamiar Aminian. Three-dimensional body and centre of mass kinematics in alpine ski racing using differential gnss and inertial sensors. *Remote Sensing*, 8(8):671, Aug 2016. 1

[9] Ross Girshick. Fast r-cnn. (arXiv:1504.08083), Sep 2015. 2

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. (arXiv:1311.2524), Oct 2014. 2

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. (arXiv:1703.06870), Jan 2018. 2, 4

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 3

[13] Christian Keilstrup Ingwersen, Christian Mikkelstrup, Janus Nørtoft Jensen, Morten Rieger Hannemose, and Anders Bjorholm Dahl. Sportspose – a dynamic 3d sports pose dataset. (arXiv:2304.01865), Apr 2023. arXiv:2304.01865 [cs]. 2

[14] Ling Li, Lin Zhao, Linhao Xu, and Jie Xu. Towards high performance one-stage human pose estimation. In *Proceedings of the 4th ACM International Conference on Multimedia in Asia*, pages 1–5, 2022. 4, 6

[15] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 4

[16] TorchVision maintainers and contributors. Torchvision: Pytorch's computer vision library. https://github.com/pytorch/vision, 2016. 4, 7

[17] Christopher Papic, Ross H Sanders, Roozbeh Naemi, Marc Elipot, and Jordan Andersen. Improving data acquisition speed and accuracy in sport using neural networks. *Journal of Sports Sciences*, 39(5):513–522, Mar 2021. 2

[18] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. (arXiv:1506.01497), Jan 2016. 2

[19] Helge Rhodin, Jörg Spörri, Isinsu Katircioglu, Victor Constantin, Frédéric Meyer, Erich Müller, Mathieu Salzmann, and Pascal Fua. Learning monocular 3d human pose estimation from multi-view images. (arXiv:1803.04775), Mar 2018. 2

[20] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

[21] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019. 2

[22] Matej Supej and H-C Holmberg. Monitoring the performance of alpine skiers with inertial motion units: Practical and methodological considerations. *Journal of Science in Sport and Exercise*, 3(3):249–256, Aug 2021. 1

[23] Huaqiang Wang, Lu Huang, Kang Yu, Tingting Song, Fengen Yuan, Hao Yang, and Haiying Zhang. Camper's plane localization and head pose estimation based on multi-view rgbd sensors. *IEEE Access*, 10:131722–131734, 2022. 2

[24] Qingyu Wang. Application of human posture recognition based on the convolutional neural network in physical training guidance. *Computational Intelligence and Neuroscience*, 2022:1–11, Jun 2022. 2

[25] Yifei Zheng and Hongling Zhang. Video analysis in sports by lightweight object detection network under the background of sports industry development. *Computational Intelligence and Neuroscience*, 2022:1–10, Aug 2022. 2

[26] Dejan Štepec and Danijel Skočaj. Video-based ski jump style scoring from pose trajectory. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, pages 682–690, 2022. 2