Optimizing Human Pose Estimation Using a Simplified UNet Architecture: An Experimental Analysis on Depth and Width Parameters

Shenghao Ren

{2252452@tongji.edu.cn}

Tongji University, No.1239 Siping Road, Shanghai, China

Abstract. Human pose estimation (HPE) is a significant problem in the field of computer vision, with wide applications in action recognition, intelligent surveillance, and other areas. With the development of deep learning, the accuracy of pose estimation has significantly improved. However, high-precision pose estimation models typically have complex network structures and high computational costs, making them difficult to apply in resource-constrained or real-time scenarios. To address this issue, this paper proposes a simple convolutional neural network named SimpleUNet based on UNet, utilizing a dataset of 2,000 athlete images and their annotated images with 14 visualized joints to perform human keypoint detection tasks. In SimpleUNet, we designed two adjustable parameters to control the depth and width of the network structure: the number of convolutional modules in the encoder and decoder, which defines the depth, and the number of channels in the network, which defines the width. We adjusted the depth from 10 to 100 in steps of 10 and the width from 1 to 9 in steps of 1, conducting a total of 90 experiments. We recorded the best model as well as information on loss, accuracy, and mIoU to analyze the relationship between the complexity of the model network and its performance in human keypoint detection. We ultimately found that moderate depth and width provide the best pose estimation performance, while excessively large or small depth and width each have their drawbacks.

Keywords: Human Pose Estimation, Human Keypoint Detection, Network Structure Adjustment, UNet, LSP Dataset.

1 Introduction

Human pose estimation (HPE) is a crucial problem in the field of computer vision, with widespread applications in human behavior analysis, action recognition, intelligent surveillance, virtual reality, and other areas. The primary goal is to accurately locate the positions of human joint points in images or videos [1].

Human pose recognition encompasses single-person and multi-person pose estimation, tracking, as well as 2D and 3D pose estimation. With advances in computer vision and deep learning technologies, human pose recognition algorithms have evolved from traditional methods to deep learning approaches [2]. Traditional methods fall into two categories: one approach involves analyzing global features and using classification or regression methods to address pose estimation problems, though these typically yield mediocre results and struggle with complex scenes, such as the template matching method proposed by Ramanan et al. [3]. The second

CONF-MLA 2024, November 21, Adana, Turkey

DOI 10.4108/eai.21-11-2024.2354631

approach is based on graph structure models, such as the Deformable Part-based Model (DPM) proposed by Felzenszwalb et al., which predicts human joint points using feature representations like HOG, Shape Context, and SIFT [4]. However, traditional methods are now largely outdated, and deep learning algorithms are more commonly used for pose estimation.

In single-person pose estimation, most current research is based on a keypoint position representation method known as a heatmap. Common algorithms include the Convolutional Pose Machines (CPM) proposed by Chen et al. [5], which iteratively refines the predicted positions of joint points to improve the accuracy of pose estimation. Based on the CPM structure, Newell et al. designed the Hourglass network, which fuses image features at different resolutions to better capture both global information and local details of human poses [6]. Multi-person pose estimation can be divided into two main categories: top-down and bottom-up. The top-down approach first detects each individual in the image using detection algorithms, then predicts the positions of joint points using single-person pose estimation methods. This approach tends to produce more accurate results. Notable methods include the Cascaded Pyramid Network (CPN) proposed by Chen et al. [7], HRNet by Sun et al. [8], and AlphaPose and RMPE (Regional Multi-Person Pose Estimation) by Fang et al. [9]. The bottom-up approach, on the other hand, first detects all body joint keypoints and then groups them into individual poses. This approach is generally faster and more suitable for handling large-scale multi-person scenes. Early methods like DeepCut [10] and DeeperCut [11], based on R-CNN, perform pose estimation through optimized segmentation and keypoint matching strategies. OpenPose, proposed by Cao et al. [12], is one of the most representative bottom-up methods currently available, predicting human keypoints and limb connections through convolutional neural networks and performing exceptionally well in multi-person pose estimation tasks. Additionally, Xiao et al. have proposed a series of improvements based on HRNet, such as HigherHRNet [13], further enhancing the accuracy and speed of multi-person pose estimation.

Currently, most high-performance pose estimation models, while capable of achieving high accuracy, have complex network structures and numerous parameters, making them less suitable for scenarios with limited computational resources or stringent real-time requirements [14]. To address this issue, this paper proposes a simplified version of the U-Net model called SimpleUNet. The U-Net structure is simple and efficient, making it particularly suitable for tasks requiring precise pixel-level output. The encoder-decoder symmetrical structure of U-Net, combined with skip connections, enables the direct transfer of shallow features from the encoder to the decoder, effectively preserving spatial information and avoiding the vanishing gradient problem. SimpleUNet reduces the network depth (H) and width (D) in a structured manner, lowering the model's complexity and computational demands while maintaining performance [15].

To validate the effectiveness of the model, this paper conducted experiments on the classic Leeds Sports Pose (LSP) dataset. The LSP dataset contains 2,000 images of various sports actions, with each image annotated with 14 keypoints. The experiments systematically explored the relationship between model complexity and performance by adjusting the depth (H) and width (D) of the model. Multiple experiments were conducted to identify the optimal model.

2 Materials

The LSP dataset is widely used as a benchmark for HPE. It contains 2,000 images of athletes collected from Flickr, an open online photo-sharing platform, using keywords related to various sports activities such as "football," "tennis," "gymnastics," and "athletics." Of these images, 1,600 are used for training and 400 for testing. The dataset covers multiple sports disciplines and includes a large number of non-daily and complex movement postures, ensuring a rich and diverse range of poses. Additionally, it encompasses various background environments, both indoor and outdoor, which helps in evaluating algorithm performance under different conditions [16-17].

Before use, the dataset was preprocessed by first reading and converting the color space through OpenCV, followed by resizing and data conversion. Specifically, all images and labels were uniformly resized to (256, 256), converted into corresponding tensor types, normalized, and standardized. Finally, any abnormal values in the labels were corrected to ensure they remained within valid ranges [18]. The data labels were manually annotated by professionals, with each image marked with 14 joint positions. From a central perspective, the left and right joints are consistently marked.

3 Method

3.1 Overall Method and Process

U-Net is a convolutional neural network proposed by Ronneberger et al. in 2015, originally designed for medical image segmentation tasks. It adopts an encoder-decoder symmetric structure, where the encoder extracts image features and the decoder gradually restores spatial information. The key to U-Net lies in its skip connections, which merge shallow features from the encoder with those of the decoder, preventing information loss from downsampling and thus excelling in detail restoration. Due to its simple and flexible structure, U-Net can be adapted to different task requirements by adjusting its depth and width, making it widely used in fields such as medical image processing and remote sensing image analysis. This adaptability makes U-Net particularly suitable for pixel-level prediction tasks like pose estimation [19].

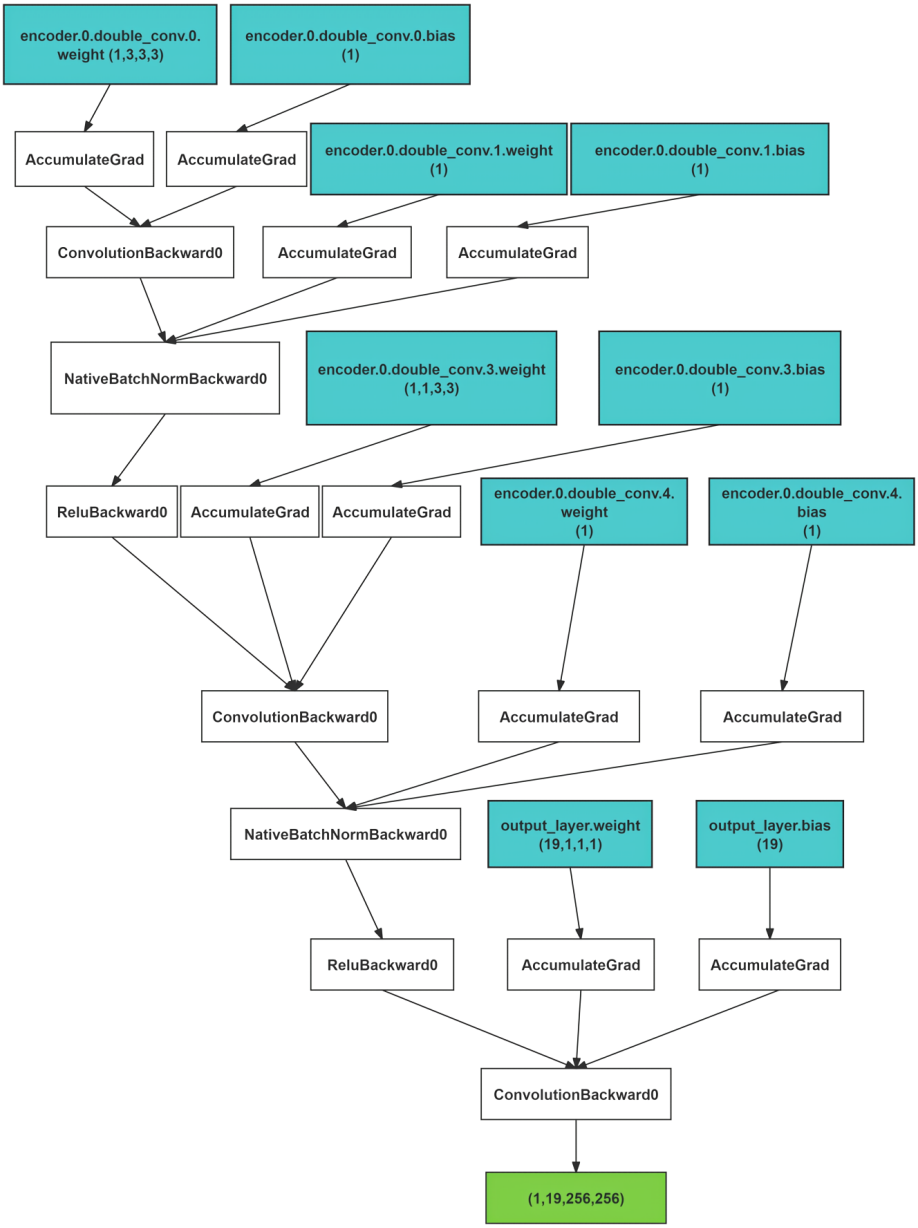

Fig. 1. SimpleUNet Network Architecture (Depth H = Width D = 1).

SimpleUNet is a simple convolutional neural network built upon the U-Net architecture, retaining its core encoder-decoder symmetrical structure. The encoder consists of H double convolution blocks, each containing two 3x3 convolution layers, batch normalization, and ReLU activation functions. The feature maps are downsampled through max pooling operations

to progressively extract deeper features. The number of channels is controlled by parameter D, which determines the output channel count for each convolutional layer. In the decoder, SimpleUNet performs upsampling through transposed convolutions (ConvTranspose2d) to restore the spatial resolution of the feature maps. At each decoding stage, the feature maps are concatenated with corresponding layers from the encoder via skip connections. The decoder also includes double convolution blocks to further process the concatenated feature maps. Finally, the model generates segmentation output through a 1x1 convolution layer, with the output channel count matching the number of target classes [20][21].

SimpleUNet was trained and evaluated on the LSP dataset for HPE tasks. By systematically adjusting the model's depth H, i.e., the number of convolutional modules in the encoder and decoder, as well as the width D, i.e., the number of channels in the network, a total of 90 experiments were conducted to analyze the impact of these parameters on model complexity and HPE performance.

The overall process begins with data preprocessing, which includes acquiring and reading the dataset, resizing all images and labels to a uniform 256x256 pixels, handling labels, and dividing the dataset with an 8:2 training-to-testing ratio. Next, we construct the model based on the U-Net design, comprising an encoder and decoder section, employing a cross-entropy loss function and the Adam optimizer with a learning rate of 0.001. By adjusting the depth H from 10 to 100 in increments of 10 and the width D from 1 to 9, models of varying complexity were created. Finally, model training and evaluation were conducted over 100 epochs, calculating loss, accuracy, and mIoU on the validation set [22].

3.2 Innovations

Traditional model optimization often focuses on network design or parameter tuning. This study, however, begins with the fundamental components of the model structure, systematically adjusting the number of control channels and the number of convolutional modules in the encoder and decoder to quantitatively analyze their impact on HPE performance [23].

The depth H determines the number of convolutional modules in the encoder, as well as the corresponding upsampling modules in the decoder. A greater depth H allows the model to extract deeper features. In the encoder, each convolutional module and pooling operation halves the feature map size [24]. The width D specifies the number of output channels for each convolutional layer, or the depth of the feature map. A greater width D enables each convolutional layer to learn and represent more feature patterns, though it significantly increases the number of model parameters and computational load, which impacts training and inference efficiency [25]. In this experiment, the depth H was adjusted from 10 to 100 in increments of 10, and the width D was adjusted from 1 to 9 in increments of 1. A total of 90 experiments were conducted, with the best model from each experiment recorded alongside metrics for loss, accuracy, and mean Intersection over Union (mIoU).

The evaluation metrics include validation loss, which measures the prediction error of the model on the validation set; accuracy, which calculates the proportion of correctly predicted joint points; and mean Intersection over Union (mIoU), which evaluates the model’s segmentation performance across various categories [26].

3.3 Experimental Details

The experiment was conducted using the LSP dataset, which consists of 2,000 images of athletes, with 1,600 images allocated for training and 400 images for validation. All images were standardized through preprocessing to ensure pixel values were distributed within the range [0,1] and resized uniformly to 256×256 pixels. Corresponding label images were resized similarly to maintain consistent resolution. To eliminate the influence of invalid categories, all label values were clipped to valid ranges, and mismatched labels were corrected.

The SimpleUNet model comprises a symmetrical encoder-decoder structure, where the depth H denotes the number of convolutional modules in the encoder and decoder, and the width D determines the number of output channels in each convolutional layer. The network's architecture is detailed as follows:

First, the encoder is composed of H sequential convolutional modules. Each convolutional module adopts a DoubleConv structure, including two convolution operations, Batch Normalization, and a ReLU activation function. These modules perform downsampling through MaxPooling, progressively halving the spatial resolution and expanding the receptive field to capture global features. The first convolutional layer of the network receives an RGB image as input, producing D output channels. For subsequent convolutional modules, the number of input and output channels is consistently D, with a kernel size of 3×3, stride of 1, and padding of 1 to ensure that the spatial dimensions of the output match those of the input. Next, the decoder consists of H−1 symmetric modules. Each decoder module includes three components: first, feature maps are upsampled using transposed convolution, doubling the spatial resolution; second, skip connections are used to concatenate the upsampled feature maps with corresponding encoder feature maps, fusing high-resolution and low-resolution features along the channel dimension; finally, the concatenated feature maps are passed through a DoubleConv module to further integrate contextual and detailed features. The decoder's output is processed through a 1×1 convolution layer to generate a prediction feature map with channels corresponding to the number of target classes. This feature map retains the same spatial dimensions as the input image, with each channel representing the pixel-wise classification probability distribution for a specific class. The output tensor has the dimensions [batch_size,num_classes,H,W], where H and W denote the height and width of the input image.

In this experiment, H was varied in the range [10,20,…,100], and D was varied in the range [1,2,…,9]. A grid search was performed over all combinations of these parameters, resulting in 90 independent experiments. The training process employed the cross-entropy loss function with the Adam optimizer, using a learning rate of 0.001. Each experiment ran for 100 epochs, during which Validation Loss, Accuracy, and mIoU were evaluated at the end of every epoch. Detailed results for each training and validation cycle were recorded in CSV files, and the best model was saved at the point of minimum validation loss.

4 Overall Evaluation

Table 1. Partial Experiment Results.

Depth H	Width D	Accuracy	mIoU
10	1	0.910151024	0.455075512
10	2	0.909503059	0.45475153
20	1	0.909738274	0.454869137
20	2	0.908985634	0.454497296
70	7	0.901004601	0.498077807
70	8	0.902648811	0.496399651
70	9	0.900734634	0.495855186
80	1	0.912134895	0.456067448
80	5	0.895676804	0.486028871
80	7	0.898684044	0.496955732
80	8	0.903111496	0.501042928
80	9	0.901546516	0.498304727
90	7	0.899385605	0.495219851
90	8	0.900071144	0.50155705
90	9	0.901426697	0.50063951
100	7	0.902854538	0.496950364
100	8	0.90153019	0.499089396
100	9	0.904422874	0.501645585

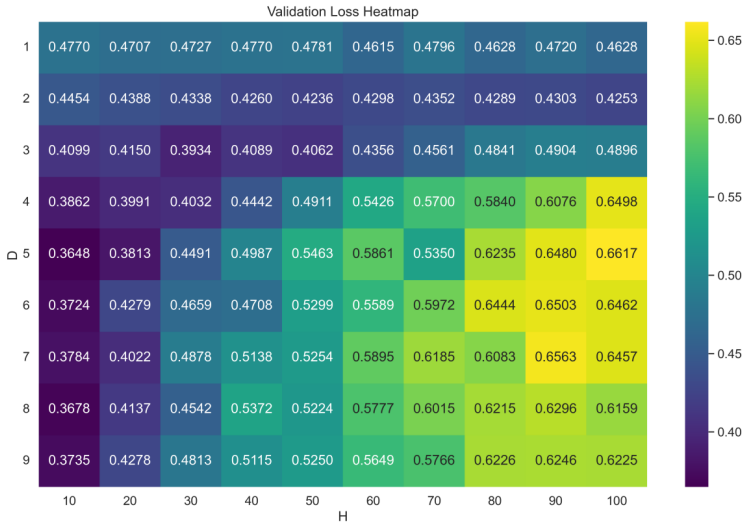

Fig. 2. Validation Loss Heatmap.

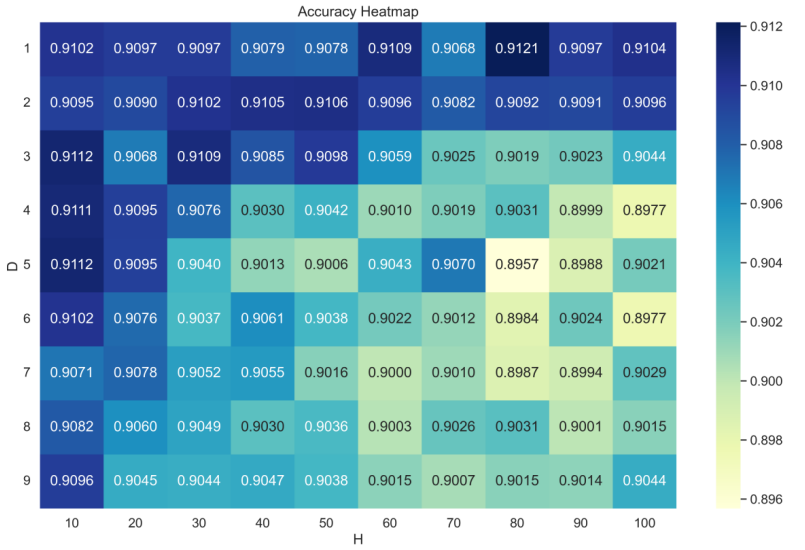

Fig. 3. Accuracy Heatmap.

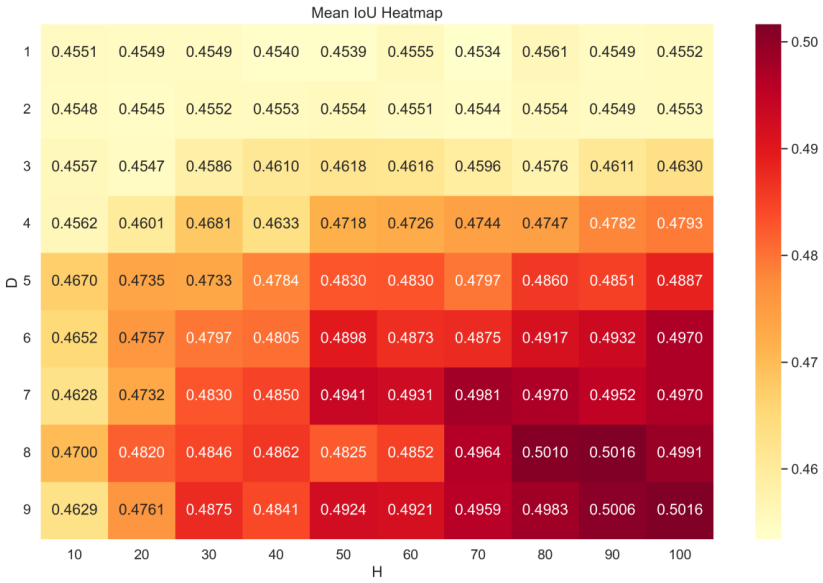

Fig. 4. Mean IoU Heatmap.

First, analyzing Figure 1, the Validation Loss Heatmap, this graph displays the distribution of validation loss across various combinations of model depth H and width D. Quantitatively, validation loss values are generally between 0.36 and 0.67. When D is between 5 and 9 and H is 10, the loss is minimized, ranging from approximately 0.36 to 0.37, with the lowest value at 0.3648 when D=5. Conversely, when D=5 and H=100, the loss is maximized at 0.6617. Overall, as H and D increase, there is an upward trend in validation loss. When H ranges from 10 to 30,

the loss decreases gradually or fluctuates as D increases. For H between 40 and 100, the loss generally rises with D; when D=1 or D=2, loss tends to decrease or fluctuate as H increases, showing minimal change. However, when D ranges from 3 to 9, the loss mainly increases with H.

Next, in the Accuracy Heatmap, the model's accuracy is mostly between 0.90 and 0.91, with slight variations. Higher accuracy values are observed when D=1, 2 or H=10, 20. The highest accuracy is achieved at D=1 and H=80, reaching 0.9121, while the lowest occurs at D=5 and H=80, at 0.8957. Overall, as D and H increase, accuracy exhibits a slight downward trend, although the changes are not substantial.

Finally, examining the mIoU Heatmap, this graph shows the average Intersection over Union (mIoU) values for model predictions across different H and D settings. The mIoU values generally range between 0.45 and 0.50. When D is between 1 and 3, mIoU remains relatively stable as H increases. In the region where D=7 to D=9 and H=70 to H=100, mIoU is relatively high, around 0.50, with the maximum value at 0.5016 when D=8, H=90, and D=9, H=100. Overall, as D and H increase, mIoU tends to rise, indicating improved segmentation performance of the model.

The parameter D represents the number of channels in each convolutional layer; the more channels there are, the more features the model can learn. When D is small, the model's feature extraction capacity is limited, making it difficult to capture complex image patterns and semantic information [27]. This limitation is especially evident in the heatmaps for validation loss and mIoU. When D=1 or D=2, the loss is higher, and the mIoU is lower, indicating that the model struggles to effectively learn data characteristics under these settings. As D increases, the model's feature extraction ability improves, allowing each convolutional layer to capture more intricate image features. An appropriate value of D helps the model adapt better to the data, enhancing segmentation accuracy [28]. However, when D exceeds 6, the model's parameter count significantly increases. Too many feature channels can lead to feature redundancy, making it difficult for the model to learn useful information from the training data, potentially resulting in overfitting. This leads to an increase in validation loss and a decrease in accuracy [29].

Shallower values of H (ranging from H=10 to H=30) result in relatively low validation loss and good mIoU performance, effectively capturing hierarchical features of the image while avoiding issues like gradient vanishing and overfitting [30]. This effect is particularly evident with smaller datasets; shallow models with a moderate number of parameters are generally easier to train. As H increases, the model becomes deeper. While deeper models can theoretically extract more abstract features, very deep networks tend to suffer from gradient vanishing during training, making them difficult to optimize [31]. Consequently, the model may not effectively utilize information learned at deeper layers, resulting in limited performance improvements. Additionally, as H increases, the number of model parameters also rises, extending training time and increasing the risk of overfitting. This is especially problematic for smaller datasets, where deep models may overfit the training data and fail to perform well on the validation set [32].

The LSP dataset is relatively small, containing only about 2,000 images. With such a small dataset, it is essential to match the model's parameter count and complexity to the dataset's scale. Overly complex models struggle to train adequately on small datasets and are prone to overfitting [33]. The increase in validation loss, along with declines in accuracy and mIoU,

indicates that when H and D are too large, the model performs well on training data but demonstrates poor generalization on validation data. This explains why, for D=6 and above or H=50 and above, the validation loss begins to rise, and the model's accuracy and mIoU do not perform as well as those of more moderate combinations of depth and width.

Based on the comprehensive analysis of these results, moderate values of H and D can maintain high accuracy and segmentation performance while significantly reducing computational resource usage, achieving an optimal balance between complexity and performance [34]. The results of this experiment provide an important reference framework for balancing computational cost and model performance by appropriately adjusting the model structure. This is particularly significant for applications requiring real-time performance, allowing the algorithm to be deployed on devices with limited computational power and expanding the application scenarios of HPE [35].

5 Conclusion

In this study, we conducted 90 experiments on a simplified SimpleUNet model, based on UNet, by adjusting the model's depth (H) and width (D). We recorded metrics such as loss, accuracy, and mean Intersection over Union (mIoU) to evaluate the model's performance and observe the impact of these two parameters on HPE using the LSP dataset.

The experiments showed that selecting appropriate values for D and H can enhance the model's feature extraction capabilities. However, excessively high D leads to increased model complexity and overfitting. When H is too large, training becomes more challenging, resulting in limited performance improvement, and in some cases, it may cause gradient vanishing or overfitting.

The model configurations with D=6-8 and H=10-30 performed best on the LSP dataset, demonstrating lower validation loss and higher accuracy and mIoU. This indicates that moderate depth and width provide optimal pose estimation performance for the LSP dataset.

Establishing an appropriate network complexity is particularly important in scenarios with limited resources or real-time requirements. The approach used in this study can be similarly applied to other research areas, exploring comparable network adjustment strategies on other datasets or tasks. By setting adjustable hyperparameters to simplify or increase network complexity, conducting multiple experiments, and analyzing and comparing results, a balance between performance and computational efficiency can be achieved.

References

[1] Chen, X., Yuan, J., Zhang, W., & Wang, C. (2020). A Survey of Recent Advances in Human Pose Estimation: The Deep Learning Perspective. IEEE Access, 8, 14695-14713.

[2] Liu, C., Shi, B., Li, J., Wang, Z., & Wang, C. (2022). A Comprehensive Survey on 2D and 3D Human Pose Estimation: Methods, Challenges, and Future Directions. ArXiv Preprint arXiv:2203.02883.

[3] Ramanan, D. (2006). Learning to parse images of articulated objects. Advances in Neural Information Processing Systems (NIPS), 19, 1129–1136.

[4] Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part-based models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(9), 1627-1645.
[5] Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., & Sun, J. (2018). Cascaded Pyramid Network for Multi-Person Pose Estimation. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 7103-7112.
[6] Newell, A., Yang, K., & Deng, J. (2016). Stacked Hourglass Networks for Human Pose Estimation. European Conference on Computer Vision (ECCV), 483-499.
[7] Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., & Sun, J. (2018). Cascaded Pyramid Network for Multi-Person Pose Estimation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 7103-7112.
[8] Sun, K., Xiao, B., Liu, D., & Wang, J. (2019). Deep High-Resolution Representation Learning for Human Pose Estimation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 5693-5703.
[9] Fang, H. S., Xie, S., Tai, Y. W., & Lu, C. (2017). RMPE: Regional Multi-Person Pose Estimation. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2334-2343.
[10] Pishchulin, L., Insafutdinov, E., Tang, S., Andres, B., Andriluka, M., Gehler, P. V., & Schiele, B. (2016). DeepCut: Joint Subset Partition and Labeling for Multi-Person Pose Estimation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4929-4937.
[11] Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., & Schiele, B. (2016). DeeperCut: A Deeper, Stronger, and Faster Multi-Person Pose Estimation Model. European Conference on Computer Vision (ECCV), 34-50.
[12] Cao, Z., Simon, T., Wei, S. E., & Sheikh, Y. (2017). Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 7291-7299.
[13] Cheng, B., Xiao, B., Wang, J., Shi, H., Huang, T. S., & Zhang, L. (2020). HigherHRNet: Scale-Aware Representation Learning for Bottom-Up Human Pose Estimation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 5386-5395.
[14] Sarafianos, N., Boteanu, B., Ionescu, B., & Kakadiaris, I. A. (2016). 3D Human Pose Estimation: A Review of the Literature and Analysis of Covariates. Computer Vision and Image Understanding, 152, 1-20.
[15] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770-778.
[16] Johnson, S., & Everingham, M. (2010). Clustered Pose and Nonlinear Appearance Models for Human Pose Estimation. BMVC.
[17] Johnson, S., & Everingham, M. (2011). Learning Effective Human Pose Estimation from Inaccurate Annotation. CVPR.
[18] Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.
[19] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), 234-241.
[20] Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., & Ronneberger, O. (2016). 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), 424-432.

[21] Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N., & Liang, J. (2018). Unet++: A Nested U-Net Architecture for Medical Image Segmentation. ArXiv Preprint arXiv:1807.10165.
[22] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
[23] Zagoruyko, S., & Komodakis, N. (2016). Wide Residual Networks. Proceedings of the British Machine Vision Conference (BMVC), 87.1-87.12.
[24] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE, 86(11), 2278-2324.
[25] Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. International Conference on Learning Representations (ICLR).
[26] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems (NIPS), 1097-1105.
[27] Tan, M., & Le, Q. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. Proceedings of the International Conference on Machine Learning (ICML), 6105-6114.
[28] Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1251-1258.
[29] Long, J., Shelhamer, E., & Darrell, T. (2015). Fully Convolutional Networks for Semantic Segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 3431-3440.
[30] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15(1), 1929-1958.
[31] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4700-4708.
[32] Srivastava, R. K., Greff, K., & Schmidhuber, J. (2015). Highway Networks. Proceedings of the International Conference on Machine Learning (ICML), 234-242.
[33] Hinton, G. E., Vinyals, O., & Dean, J. (2015). Distilling the Knowledge in a Neural Network. ArXiv Preprint arXiv:1503.02531.
[34] Ma, N., Zhang, X., Zheng, H.-T., & Sun, J. (2018). ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. Proceedings of the European Conference on Computer Vision (ECCV), 122-138.
[35] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., … & Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. ArXiv Preprint arXiv:1704.04861.