Human pose estimation using contrastive learning

Alessia Pivotto, Davide Cavicchini, Sofia Lorengo

Abstract

This study involves the use of contrastive learning techniques, more specifically SimSiam and SimClr, for human pose estimation. Various architectural models were used to understand which one had the best impact on the final performance. The setups used differed based on the number of layers; the ones tested were 1, 2, and 3-layer configurations. Results reveal that SimCLR consistently outperforms SimSiam across experiments, with SimCLR demonstrating superior feature extraction capabilities and proficiency in capturing nuanced pose details. The evaluation extends to a 3D regression task, providing insights into the models' abilities to learn invariant representations of poses from different perspectives. The analysis highlights potential areas for improvement and suggests avenues for refining contrastive learning in human pose estimation.

1 Introduction

Human pose estimation is a computer vision task with applications in several fields. This research is based on a 2D human pose estimation project that uses two different contrastive learning techniques namely SimCLR and SimSiam. Our study extends the previous investigation into 3D human pose estimation. This transition introduces new challenges and opportunities for improvement. In this study, we analyze various model configurations, including 1-layer, 2-layer, and 3-layer architectures. In the evaluation, we aim to include both qualitative and quantitative assessments. We also try to retrain the base encoders from scratch to experiment with them and discover valuable insight into promising directions. The results highlight the effectiveness of contrastive learning for 3D human pose estimation, offering a foundation for further investigation in this direction.

2 Previous work

Before delving into our 3D human pose estimation project, it is necessary to conduct a succinct review of the project "Contrastive human pose pre-training" for 2D human pose estimation by Olha Khomyn, which laid the groundwork for our project. For the contrastive pre-training, she used two different approaches: SimCLR and SimSiam.

2.1 models

SimCLR consists of a convolutional neural network base encoder (ResNet50 in our case) with a projection head and a contrastive loss function. The model was trained for 20 epochs using a batch size of 200, a learning rate of 0.01 for the projection head, and 0.01/2 for the base encoder. In the end, she obtained the following silhouette scores: 0.45 for the encoder features and 0.53 for the projection head. The obtained clusters projected using PCA can be seen in A1.

SimSiam uses the same ResNet50 network followed by the same projection head and a prediction head. Since this model does not use negative examples, it does not rely so heavily on the batch size. This model was trained for 20 epochs with batch size of 256 and a learning rate of 0.05 for all layers. As a result, she obtained the following silhouette scores: 0.48 for the encoder features, 0.53 for the projection head features, and the same for the prediction head features. The obtained clusters projected using PCA can be seen in A2.

2.2 Comparison

Upon examining the silhouette scores of both models, a near-equivalent performance is evident. However, an assumption can be made in favor of the SimCLR method for its ability to yield commendable results even with a modest batch size. However, we will discuss its limitations based on the results we expose in section 4. We prospect a noteworthy performance boost with an increase in batch size, and thus of negative pairs. Furthermore, a random visualization of images within clusters revealed a discernible pattern wherein poses demonstrated heightened similarity compared to those extracted by the SimSiam method. It is important to acknowledge that this observation may be influenced by the specific examples chosen for visualization.

Criticality may be uncovered by projecting the clustering results of SimCLR using LDA as shown in figure A16, where we can see the clusters to be compact and further apart from each other. This is favorable for clustering, but we argue that this might not be favorable for a regression task; we'll look into this with more details in section 6. This critique also extends to the use of the silhouette score as a performance metric for the contrastive learning pretraining, since it favours those exact properties.

To comprehensively compare the models for the application to human pose estimation, a final linear regression training was conducted. This involved freezing all pre-trained layers of the models, removing the projection heads, and substituting them with a linear layer. The models were then subjected to supervised training using L1 loss, where the objective was to predict the 2D coordinates of 19 joints. Both models underwent training for 20 epochs, utilizing a learning rate of 0.05 and a batch size

of 128. The dataset was partitioned into three segments, allocating 60% for training, 20% for evaluation, and another 20% for testing.

For evaluation purposes, the average Euclidean distance was adopted as the primary metric that serves as a robust indicator of model performance in capturing the spatial relationships between predicted joints.

Table 1: Evaluation metric results

Model	Training	Validation	Test
SimCLR	0.4431	0.4442	0.4440
SimSiam	0.4075	0.4098	0.4111

After training, the SimSiam model demonstrated slightly superior results in terms of average Euclidean distance. However, a nuanced observation during joint visualization revealed a distinctive characteristic: SimSiam tends to center joints along a line, a pattern not observed in joints produced by SimCLR. Both models showcased commendable performances across clustering and human pose estimation tasks. Noteworthy improvements could be achieved by extending training duration and/or utilizing larger mini-batches.

3 Metodologies

To accommodate the existing code to our task (3d regression), we focused on modifying the loss function and the final model structure. The loss function itself computes a simple MSE but, since we want to see if the contrastive learning pre-training extrapolated an invariant representation of the pose to the angle from which it is viewed, we had to calculate the rotation of the predicted joints to best fit the ground truth. More detailed specifications will be given below.

3.1 Loss

To achieve the best possible rotation, a rigid transformation known as Euclidean isometry needs to be calculated. This transformation ensures that the distance between every pair of points remains the same, thereby preserving both the shape and size of the object. The process for determining this rotation can be summarized as follows:

1. Centroid Centering: Obtain centroids for each set of points. Then center each set around them.
2. Singular Value Decomposition (SVD): Employ Singular Value Decomposition to the covariance matrix.
3. Rotation Matrix Calculation: Utilize the SVD to calculate the rotation matrix.
4. Determinant Check: Confirm the correctness of the rotation matrix by incorporating a determinant check, reflecting the matrix if the determinant is found to be negative.

To save on computation we wanted to skip the centroid centering, for this reason, we assume joint 0 to be at (0,0,0). Therefore, the model doesn't need to predict it and only outputs a vector of 18*3 values, which are then grouped into the three dimensions of each joint. To account for this, an additional point (0, 0, 0) corresponding to the neck is added in the loss function. Afterward, the ground truth is also centered around the first joint. Once this is done, the rotation matrices are determined for each batch and applied to them. Finally, the Mean Squared Error (MSE) is calculated to conclude the computation of the loss function. In terms of accuracy, we kept the average Euclidean distance as a metric.

3.2 Training setup

For training, we employed a cosine annealing learning rate scheduler coupled with the Adam optimizer. The training process spanned 30 epochs, with an initial learning rate of 0.02 and a batch size of 128. The choice of Adam optimizer was motivated by its effectiveness in handling sparse gradients, as indicated by the presence of many zeros encountered in the encoded representation of the images, and its general suitability for regression tasks. As per the division of the dataset, we kept it as it was allocating 60% for training, 20% for evaluation, and another 20% for testing.

4 Results

In this section we gather all the results from the different experiments we conducted on the pre-trained models of ResNet50 obtained using SimClr and SimSiam techniques.

4.1 1-layer

For the first set of experiments, we utilized SimCLR and SimSiam with a linear layer of dimensions $2048 \rightarrow 18 * 3$ replacing the fully connected layers (projection heads). The final model performance measures are the following:

Table 2: Results 1-layer

	Loss			Accuracy		
Model	training	validation	testing	training	validation	testing
SimCLR	19.14068	19.46107	19.639	5.0542	5.0586	5.0599
SimSiam	243.45626[1]	205.13122	233.51521	4.8203	4.8210	4.8263

[1]strangely discrepant with the results during training

The results for each training epoch can be seen in figure A3.

As we can see SimSiam was not able to work well with the validation set and even very small adjustments at the end of the training threw off both validation and even training loss results. SimCLR on the other hand, seems to be able to extract relevant information from the representation obtained by the contrastive learning step.

In the appendix are some hand-picked examples from the test set to show the models performance for SimCLR A6 and SimSiam A7. As we can see, SimCLR learned to predict symmetric poses, as we can see from the examples having both arms and legs move together. However, it was not able to learn a good representation for the movement of a single arm or leg. SimSiam has a similar behavior but a lot more exaggerated since it mostly learned a "median" representation of the poses in the dataset.

4.2 2-layers

In the subsequent experiments, we introduced an additional fully connected layer, resulting in a regression head structure of $2048 \rightarrow 128(ReLU) \rightarrow 128 \rightarrow 18 * 3$. The final model performance measures are the following:

Table 3: Results 2-layer

	Loss			Accuracy		
Model	training	validation	testing	training	validation	testing
SimCLR	16.60539	14.88832	14.95841	5.0327	5.0404	5.0415
SimSiam	22.31789	23.84964	24.22212	4.6193	4.6262	4.6276

The results for each training epoch can be seen in figure A4.

As before using the examples from the test set we obtain the predictions shown in figure A8, we see that SimCLR was able to obtain a good performance, suggesting that the representation generated by the contrastive pretraining is playing a strong role. SimSiam this time was able to learn a regression model that makes use of the representation learned as showcased in its predictions in figure A9. But it's still inferior to SimClr, it does score better in accuracy but it does worse with the MSE loss, meaning that for cases less prevalent in the data it makes bigger mistakes.

However, as before, we can see from the examples in the test set that both SimCLR and SimSiam still have the same tendency to predict symmetric poses, more accentuated in the latter.

4.3 3-layers

Expanding further, we incorporated a three-layer fully connected architecture with dimensions $2048 \rightarrow 512(ReLU) \rightarrow 512 \rightarrow 128(ReLU) \rightarrow 128 \rightarrow 18 * 3$. Despite the increased complexity and the longer training of 40 epochs, the results did not exhibit substantial improvements over the two-layer configuration. This suggests that, for our specific task, the additional layer did not contribute significantly to enhancing the model's performance. In this experiment, SimSiam was terminated early due to its oscillating pattern during early training and previous results. For this reason, here are only the final model results for SimCLR:

Table 4: Results 3-layer

	Loss			Accuracy		
Model	training	validation	testing	training	validation	testing
SimCLR	16.61583	17.11332	17.20006	5.1129	5.1174	5.1187

The results for each training epoch can be seen in figure A5. Although the model had a longer training time, the loss curves suggest that the learning rate used is not optimal, and thus the model itself could be further trained to improve performance.

Looking at the test data in figure A10, it exhibits very similar behaviors to the previous models.

5 In-depth analysis

In this section, we will explore in detail the representation learned by the ResNet50 model. We'll start by analyzing the specific transformations that characterize the first convolution layer in the network. Then, we will extend our gaze to the next layers, trying to find some key contributions of each level without ignoring the importance of increasing complexity in the representation path.

At each inference, the first layer of the neural network takes in three images, resulting from breaking down the image into its RGB channels. The network then applies various filters to the images to obtain 64 different images each hopefully having relevant information for our task. We can see all the representations obtained using three different test images in A11. Highlighted in red, purple and green are the images on which we will analyze some of the applied filters. Specifically, we'll see how red refers to a high pass filtering, purple to a low pass filtering and green to a negative photographic effect.

As we move towards the intermediate layers of the network, the representations become more complex and abstract. This phenomenon can be observed in figure A15. The images from the filters of the first layer undergo further manipulation, which is hard to comprehend by humans. The analysis of these images becomes increasingly intricate, due to the connections between the feature maps that make it difficult to immediately decompose them visually. These interconnected layers act as feature pullers, isolating significant details and patterns, and ultimately leading to a more profound understanding of images.

5.1 High pass

Using the comparisons shown in figure A12, we can see that the images were the result of applying high-pass filtering. This processing approach aims primarily to accentuate the silhouettes, focusing attention on the key elements of the image. This filtering technique can be considered a valuable tool in the field of image manipulation, useful to improve the visual definition and highlight particular details.

Analyzing the kernel of the high pass filter for the red channel, shown below, we

see it's designed to emphasize rapid image changes and reduce low-frequency components. This qualitative analysis is supported by an almost zero-sum of its components (0.0071), and having negative values in the edges and positive at the center.

$$\begin{bmatrix} 0.0023 & -0.0321 & -0.0039 & -0.0125 & -0.0851 & -0.0224 & 0.1448 \\ 0.0298 & 0.0104 & -0.0041 & -0.0536 & -0.3131 & -0.3287 & 0.0601 \\ 0.0111 & 0.0092 & 0.1226 & 0.1340 & -0.3704 & -0.5951 & -0.2644 \\ 0.0573 & 0.0649 & 0.3319 & 0.6767 & 0.2513 & -0.2713 & -0.2088 \\ -0.0967 & -0.1151 & 0.1277 & 0.4881 & 0.4082 & -0.0066 & -0.0822 \\ -0.0856 & -0.2264 & -0.0531 & 0.2177 & 0.2357 & 0.0021 & 0.0172 \\ 0.0532 & -0.1861 & -0.1513 & 0.0275 & 0.1330 & -0.0254 & -0.0156 \end{bmatrix}$$

5.2 Low pass

Similarly, using the comparisons shown in figure A13, we can see that the images are probably being processed through the application of a low-pass filter because of its distinctive features. The low-pass filter acts by attenuating the higher frequencies and accentuating the lower ones, producing a blurring effect or gradual damping. Compared to the original image, a uniform distribution of shades and light transitions is observed, reducing the presence of finer details. This technique is often used to eliminate noise or to achieve a more uniform effect in the image, being particularly useful in contexts where you want to emphasize visual consistency and minimize the disturbing elements. The use of the low pass filter can be strategic in situations where clarity and detailed sharpness are not a priority.
Analyzing the kernel for the red channel, shown below, we see it's designed to average the values of the pixels, eliminating high-frequency components. This qualitative analysis is supported by the positive sum of its components (2.4751), and its similarity with the values in a Gaussian kernel with higher values at the center and lower ones on its edges.

$$\begin{bmatrix} 0.0269 & -0.0692 & -0.0056 & 0.0635 & -0.0635 & -0.0081 & 0.0346 \\ 0.0040 & -0.0071 & 0.1616 & 0.2279 & 0.0785 & 0.0396 & -0.0044 \\ -0.0517 & 0.0185 & 0.2755 & 0.3411 & 0.1082 & 0.0609 & 0.0029 \\ -0.0407 & 0.0345 & 0.2471 & 0.2760 & 0.0752 & 0.0241 & 0.0355 \\ -0.0399 & -0.0210 & 0.1453 & 0.1981 & 0.0657 & -0.0254 & -0.0307 \\ 0.0087 & -0.0295 & 0.0759 & 0.1380 & 0.0080 & -0.0373 & -0.0433 \\ 0.0190 & -0.0311 & 0.0251 & 0.1204 & 0.0474 & -0.0036 & -0.0005 \end{bmatrix}$$

5.3 Negative

While using the comparisons shown in figure A14, we can see that the network used a kernel that resulted in the reversal of the brightness values, producing a negative photographic effect. Previously dark areas are now illuminated, while lighter areas have become darker. The reversal of the histogram can lead to a redefinition of the visual characteristics of the image, offering new interpretations and perspectives.

6 Promising directions

To further test the potential of a contrastive pretraining for human pose estimation we trained from scratch, using ResNet50's pre-trained weights, a SimCLR network in 22 epochs, and batch size of 225. But, instead of using a fixed learning rate as explained in 2.1, we opted for a cosine annealing scheduler with an initial LR of $0.3 \times batch\_size/256$, the same formula used in the original paper "A Simple Framework for Contrastive Learning of Visual Representations", and T_max 25.

Its performance based on the silhouette score seems to be worse than the previous models: 0.30 for the encoder features and 0.48 for the projection head features. However, as we already discussed in section 2.2, we think that this score correlates poorly with the results of the pose estimation task. Clusters still have discernible patterns but the different poses are further apart from each other. In addition, a random visualization of images revealed good properties for the encoding space projected in 3 dimensions using LDA A17, such as arm height correlating to one of the axes giving up hope for using this representation in pose estimation. As before, it is important to acknowledge that this observation may be influenced by the specific examples chosen for visualization.

We then trained the final regression model using the best performing configuration of 2 layers, as showcased in section 4, and the same training setup (3.2). The results are quite promising:

Table 5: Results regression with re-trained SimCLR

	Loss			Accuracy		
Model	training	validation	testing	training	validation	testing
SimCLR	10.07128	10.60058	10.68683	4.6568	4.6639	4.6647

The results for each epoch can be viewed in A18. This model surpasses its predecessors and exhibits better performance in predicting the independent position of an individual arm or leg, as demonstrated in the test images A19. However, this behavior is more apparent in the earlier epochs, which suggests that we might be overfitting the dataset. Nevertheless, it is still significant since it was not observed in the previous model.

7 Known issues

While observing the results of the models using the test set, we noticed some acquisition errors in the dataset concerning, for example, images 171204_pose2;hd_00_20 person 5,6,7 that appeared to be entirely green. Since we didn't notice this problem early enough in the training of the models, all results are obtained including those corrupted images.

We don't know how much they influenced the contrastive learning step or the regression step, but we wanted to point it out.

8 Conclusions

In summary, our investigation into contrastive learning techniques for human pose estimation underscores the superior performance of SimCLR compared to SimSiam. Across various experiments, SimCLR consistently outperformed SimSiam in capturing nuanced features related to arm height, leg positions, and arm configurations. The three-layer configuration did not yield substantial improvements, suggesting that additional complexity did not significantly help in exploiting the representation learned by the contrastive learning step.

The evaluation of SimCLR's performance across different architectures revealed its proficiency in extracting relevant information from contrastive pretraining, showcasing improved results with the introduction of additional layers. On the contrary, SimSiam faced challenges in adapting to the validation set, and even minor adjustments during training led to drastic discrepancies in results. Notably, both SimCLR and SimSiam exhibited limitations in capturing the movement of individual arms or legs, leaning towards learning median features.

Upon analyzing our findings, it is evident that the pre-trained models tend to prioritize average poses, potentially overlooking the less common ones. As mentioned in section 2.2, this could be attributed to the use of a small batch size in the SimCLR model, leading to an ineffective use of the encoding space. This observation is also applicable to the SimSiam model, which does not utilize them at all. Another possible explanation for this issue could be the absence of an appropriate learning rate scheduling, as emphasized in section 6 since our model outperformed previous works.

Although our study shows promising results, it is essential to address known issues in the dataset in future projects. Furthermore, fine-tuning the LR scheduling and addressing potential overfitting concerns in the retrained model could further improve its ability to generalize.

Overall, our investigation provides a foundation for future studies on contrastive learning for human pose estimation, offering valuable insights into its strengths and areas that require refinement.

Appendix A Images

Fig. A1: PCA visualization of clusters for SimCLR

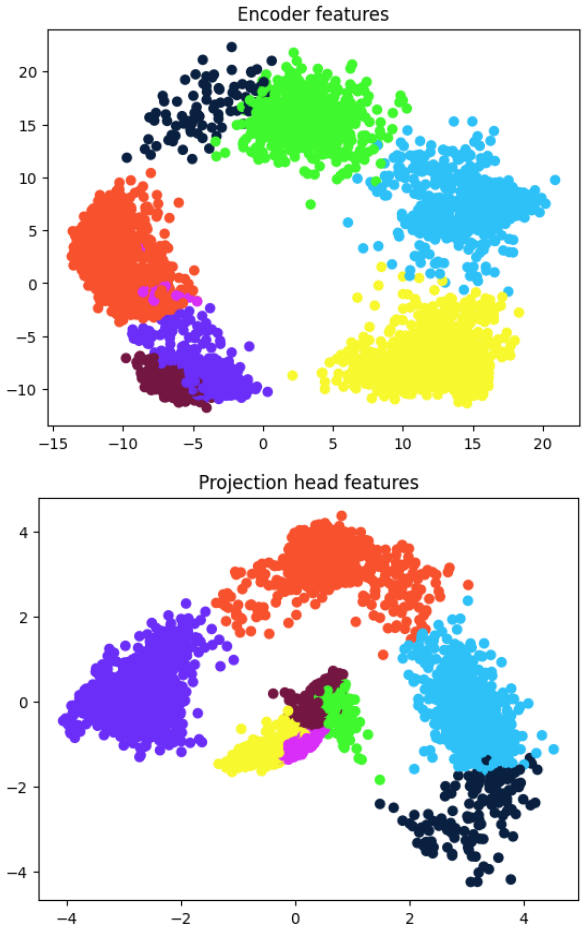


Fig. A2: PCA visualization of clusters for SimSiam

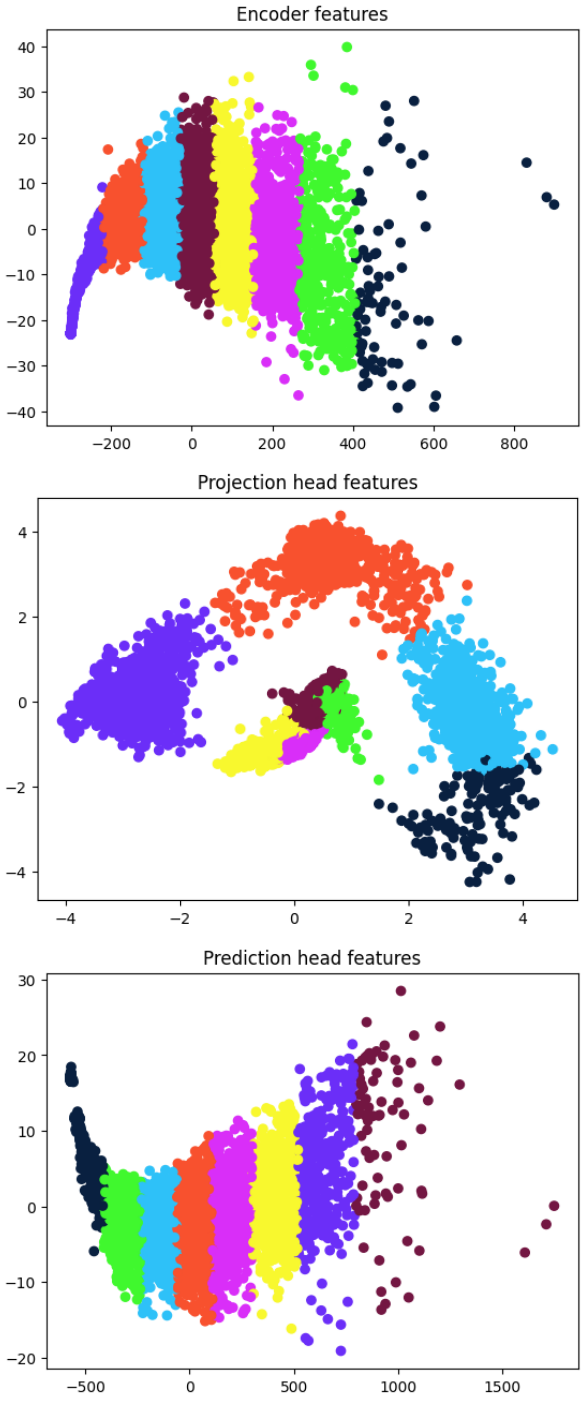
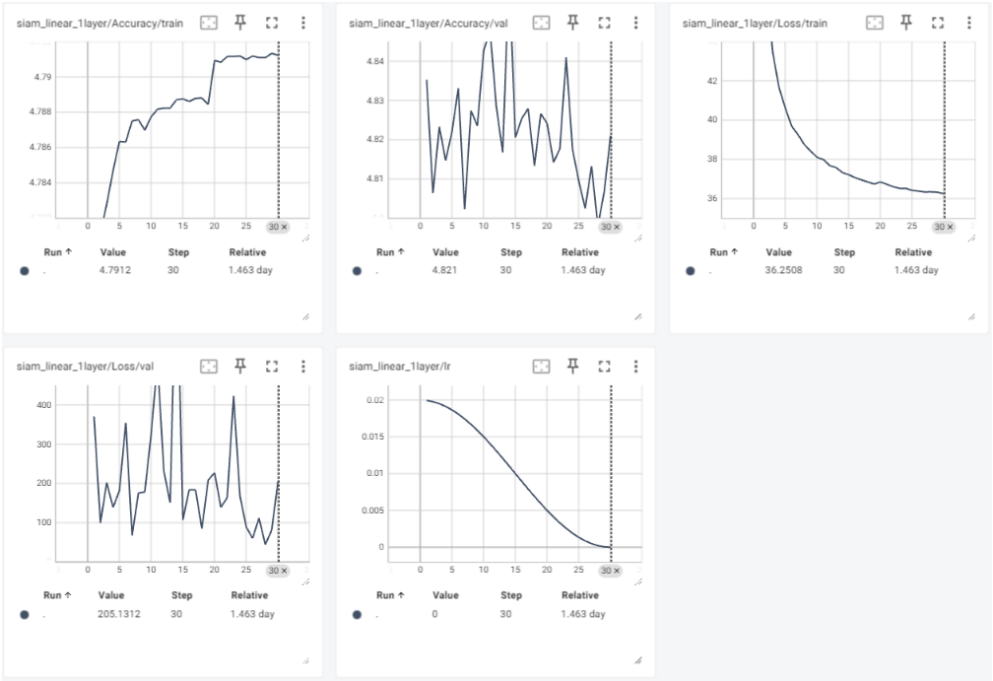


Fig. A3: results 1-layer 3d pose estimation

(a) SimSiam Results



(b) SimCLR results

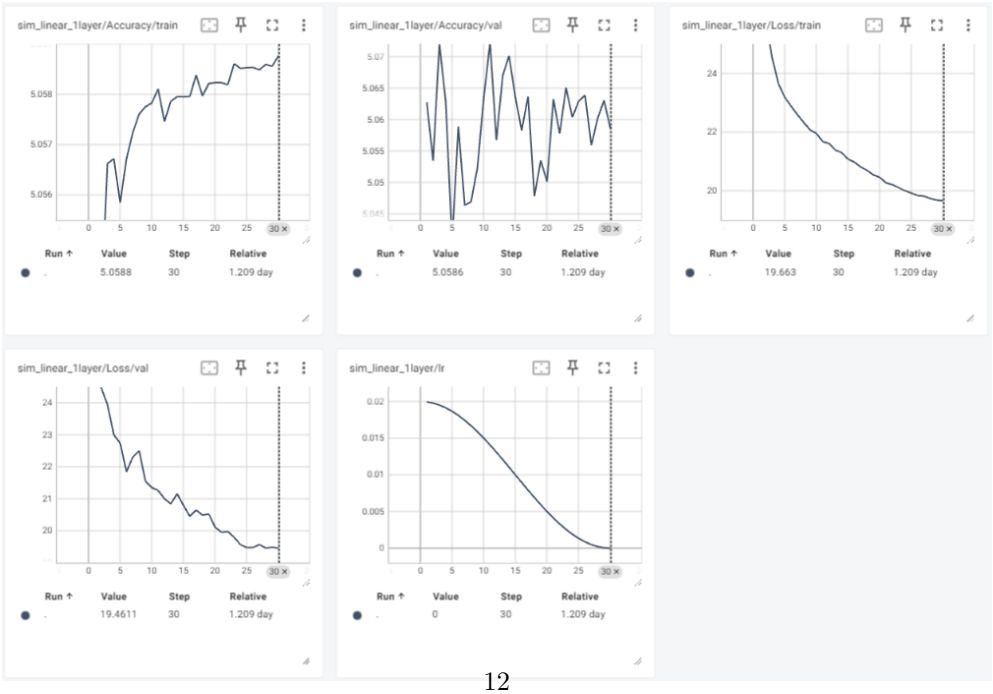
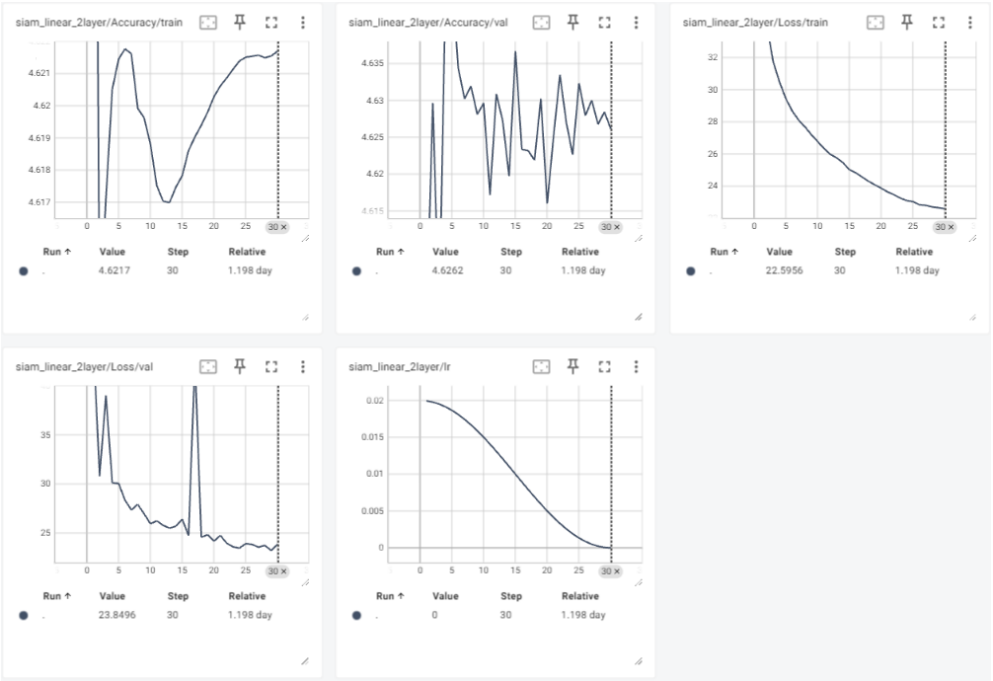


Fig. A4: results 2-layer 3d pose estimation

(a) SimSiam Results



(b) SimCLR results

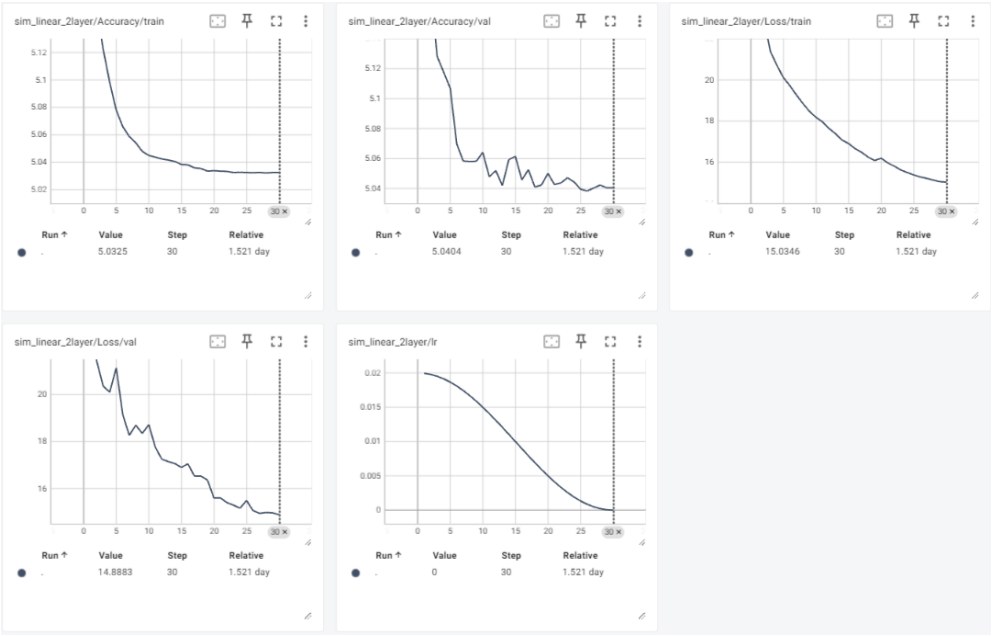


Fig. A5: results 3-layer 3d pose estimation

(a) SimCLR results

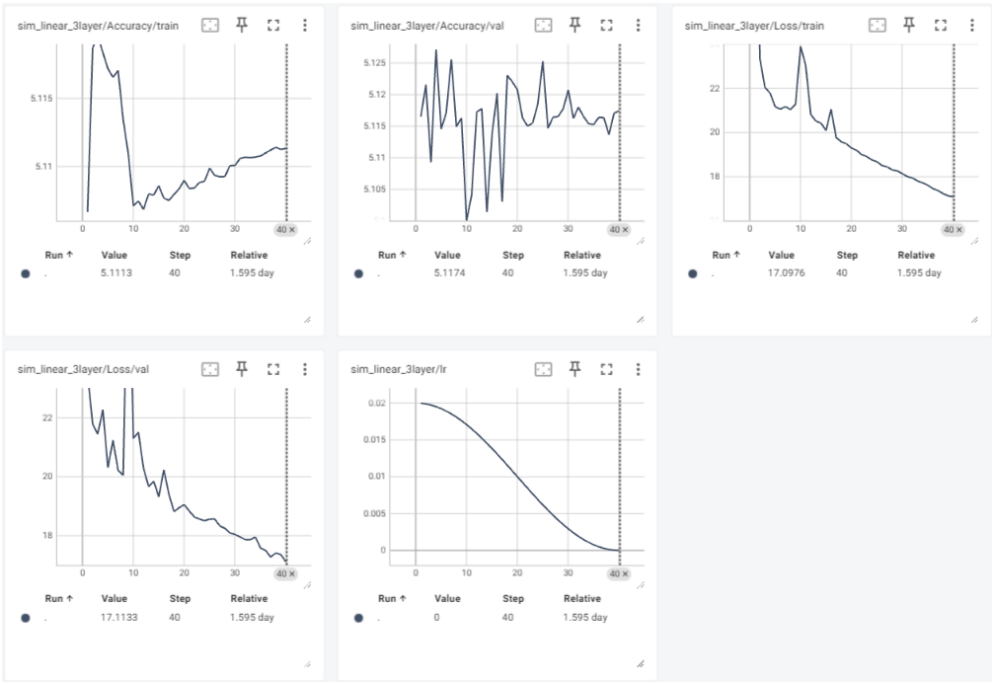


Fig. A6: 1 layer SimCLR predictions

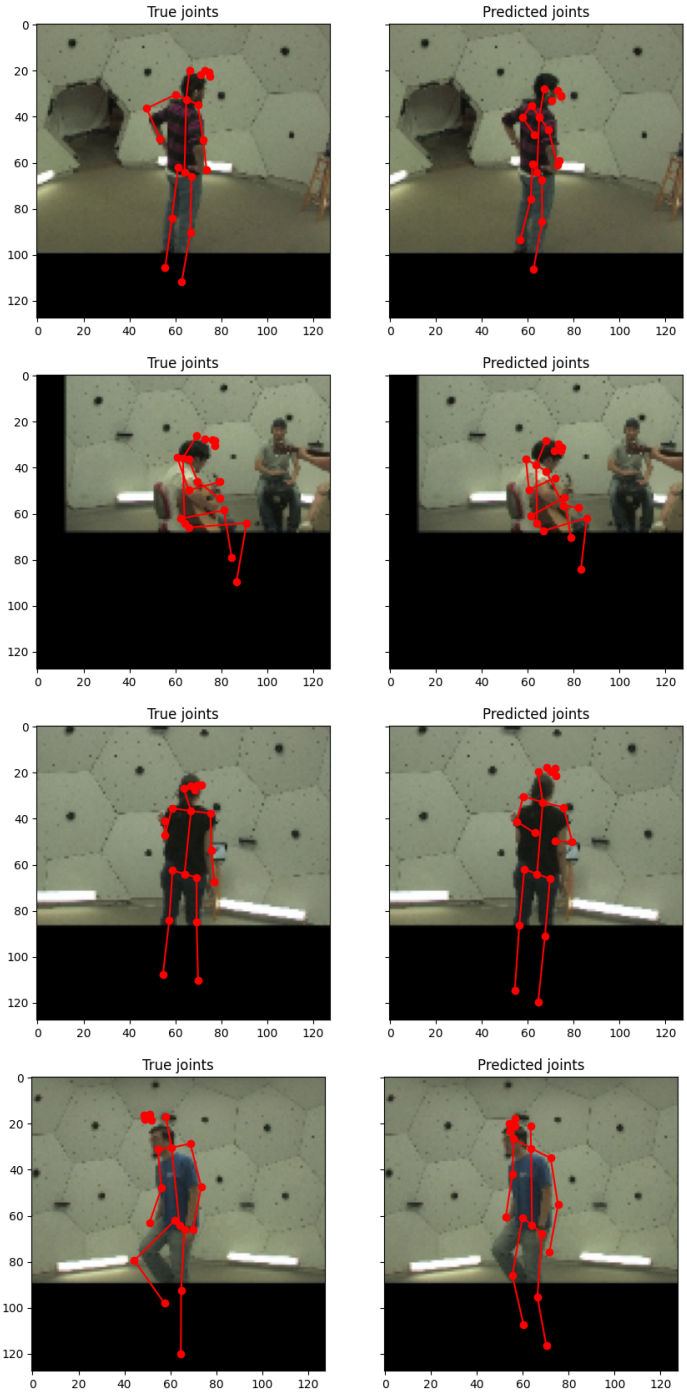


Fig. A7: 1 layer SimSiam predictions

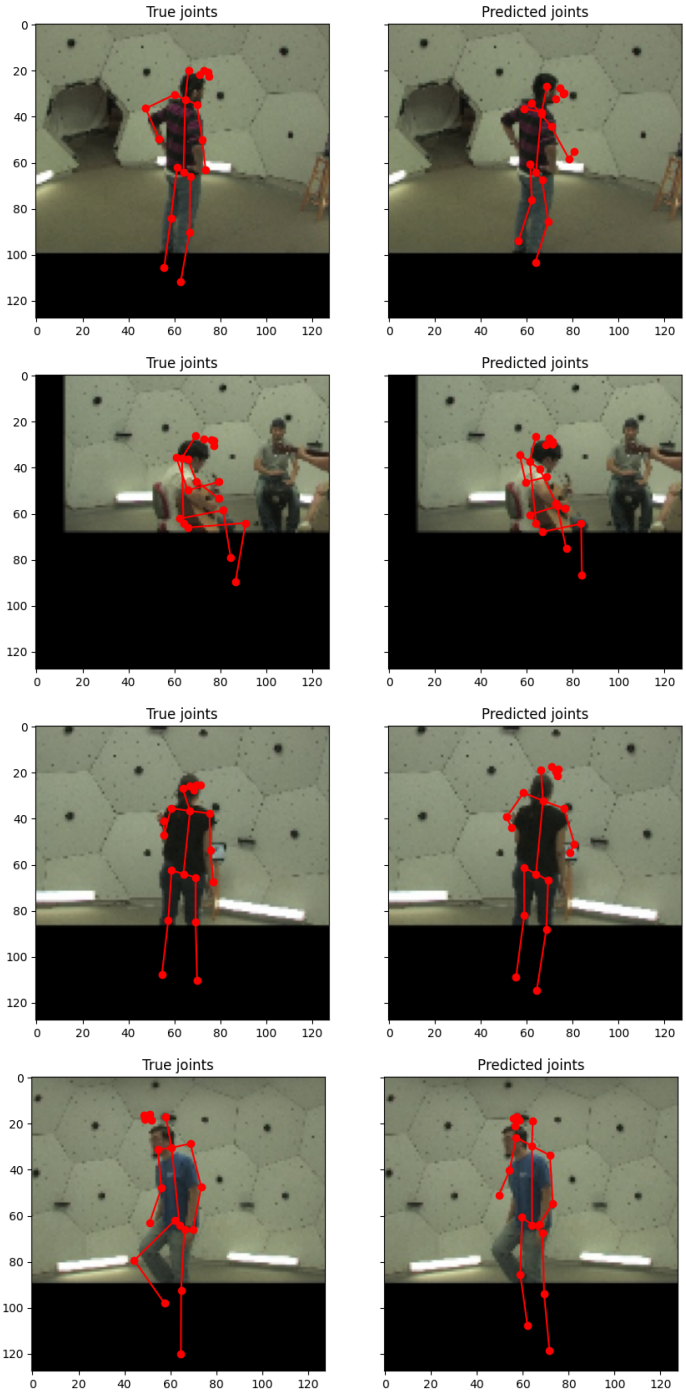


Fig. A8: 2 layer SimCLR predictions

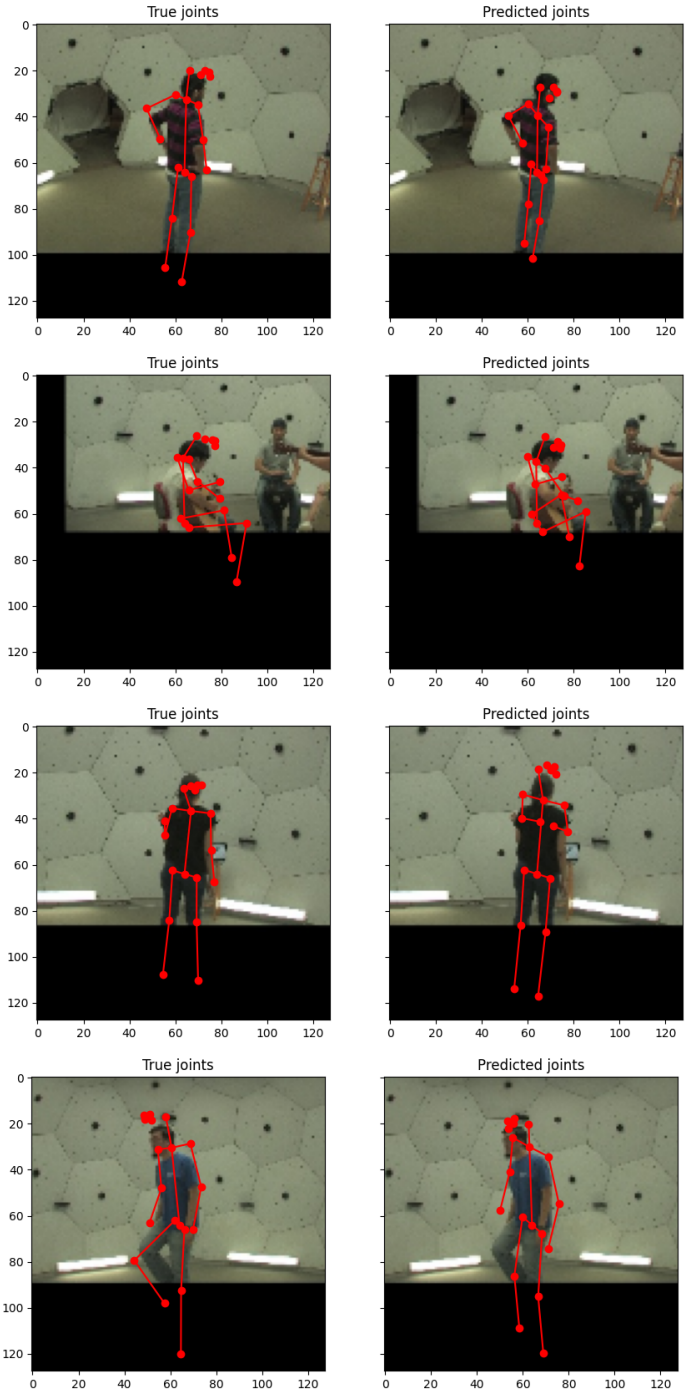


Fig. A9: 2 layer SimSiam predictions

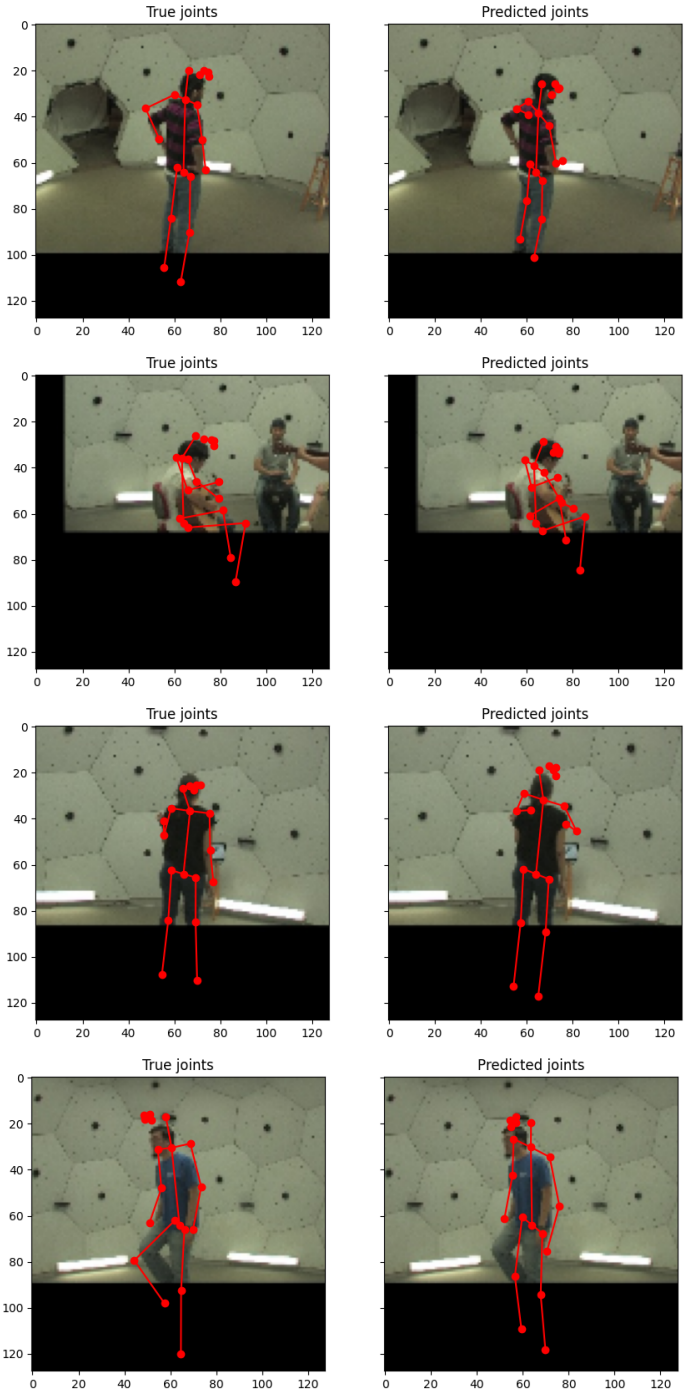


Fig. A10: 3 layer SimCLR predictions

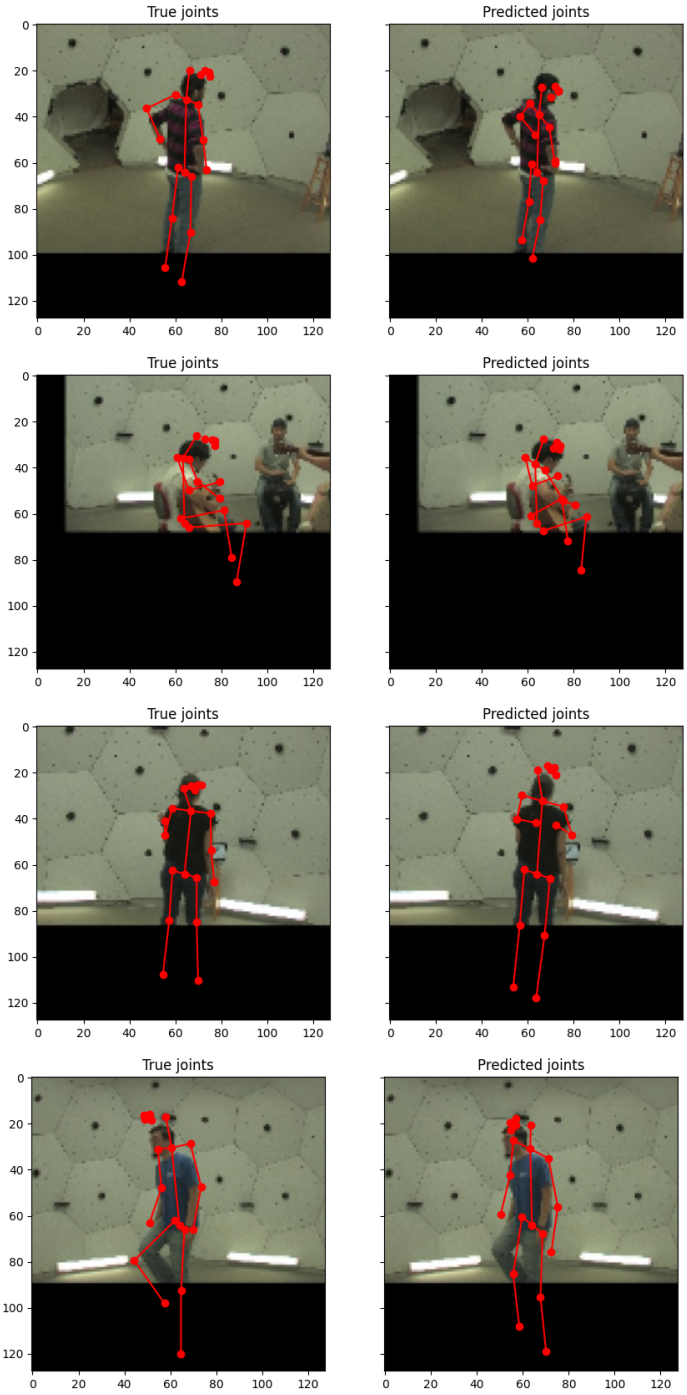


Fig. A11: Intermediate layer representations conv1

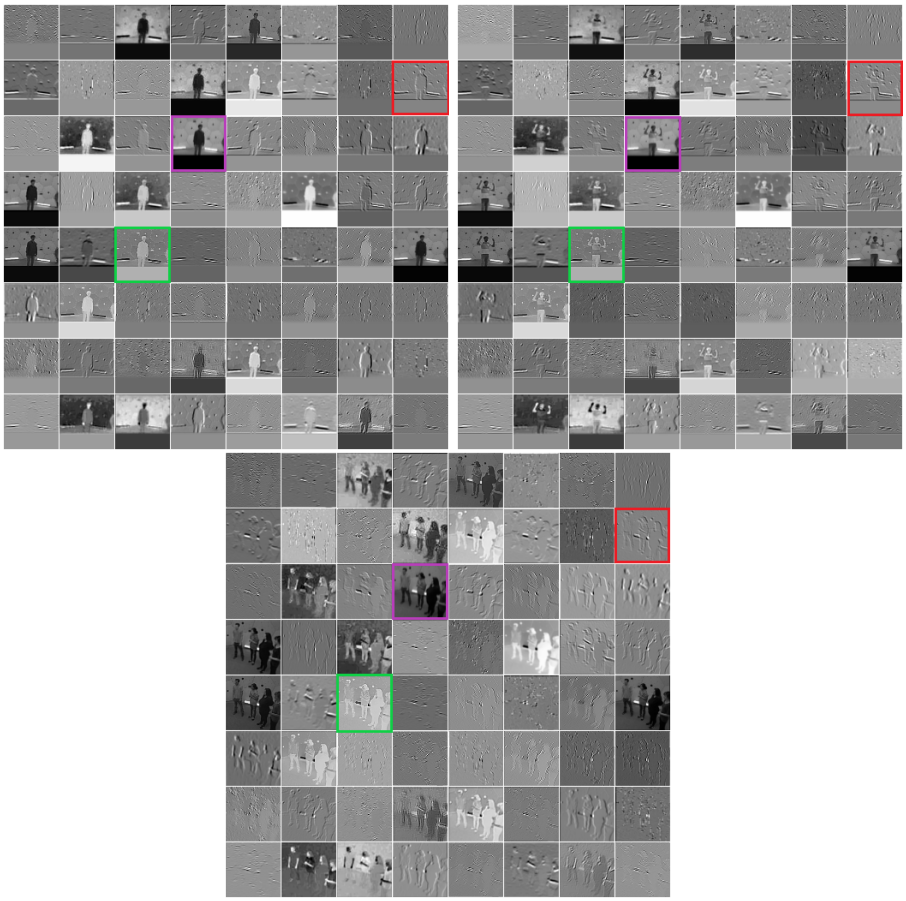


Fig. A12: Comparison HPF

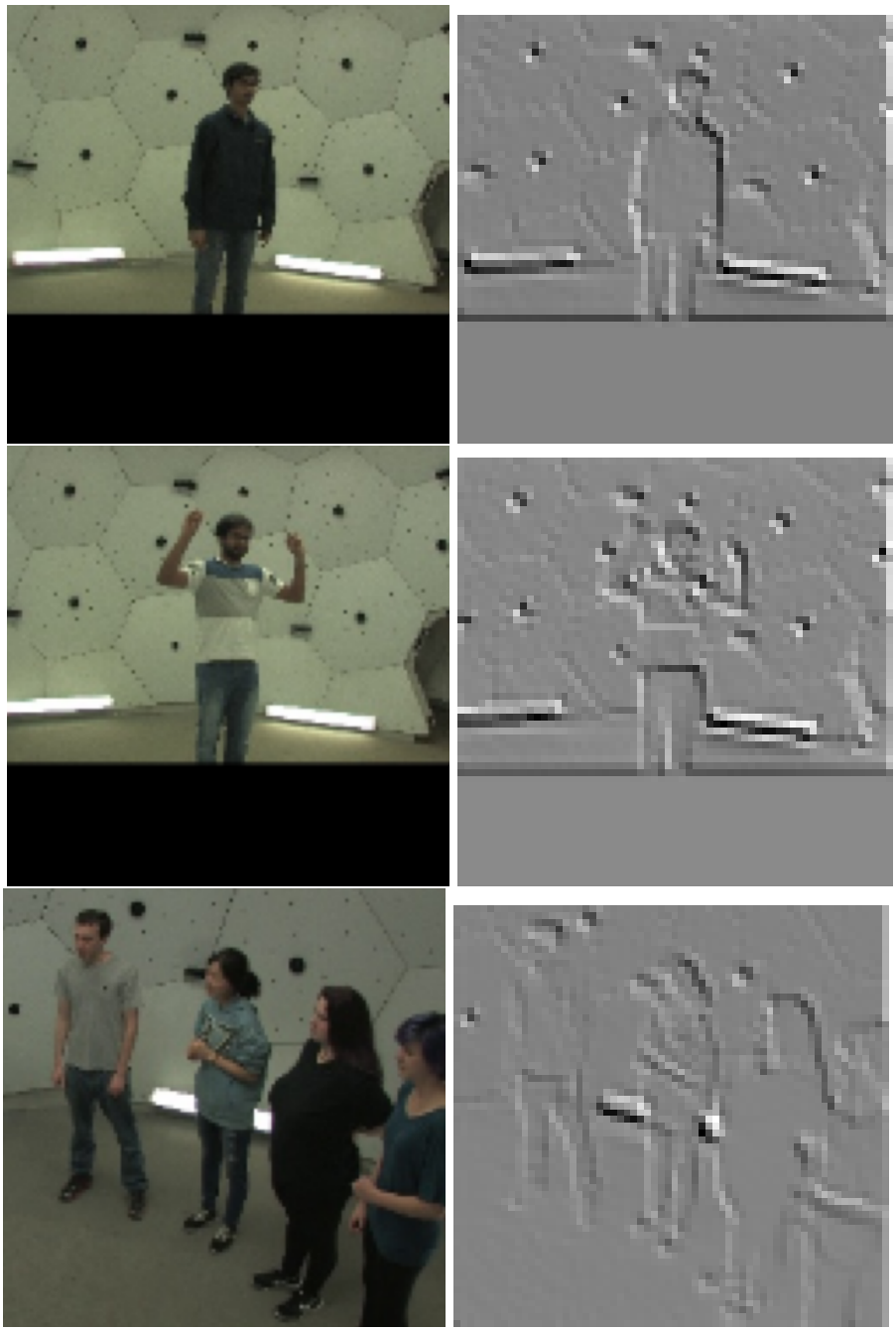


Fig. A13: Comparison LPF

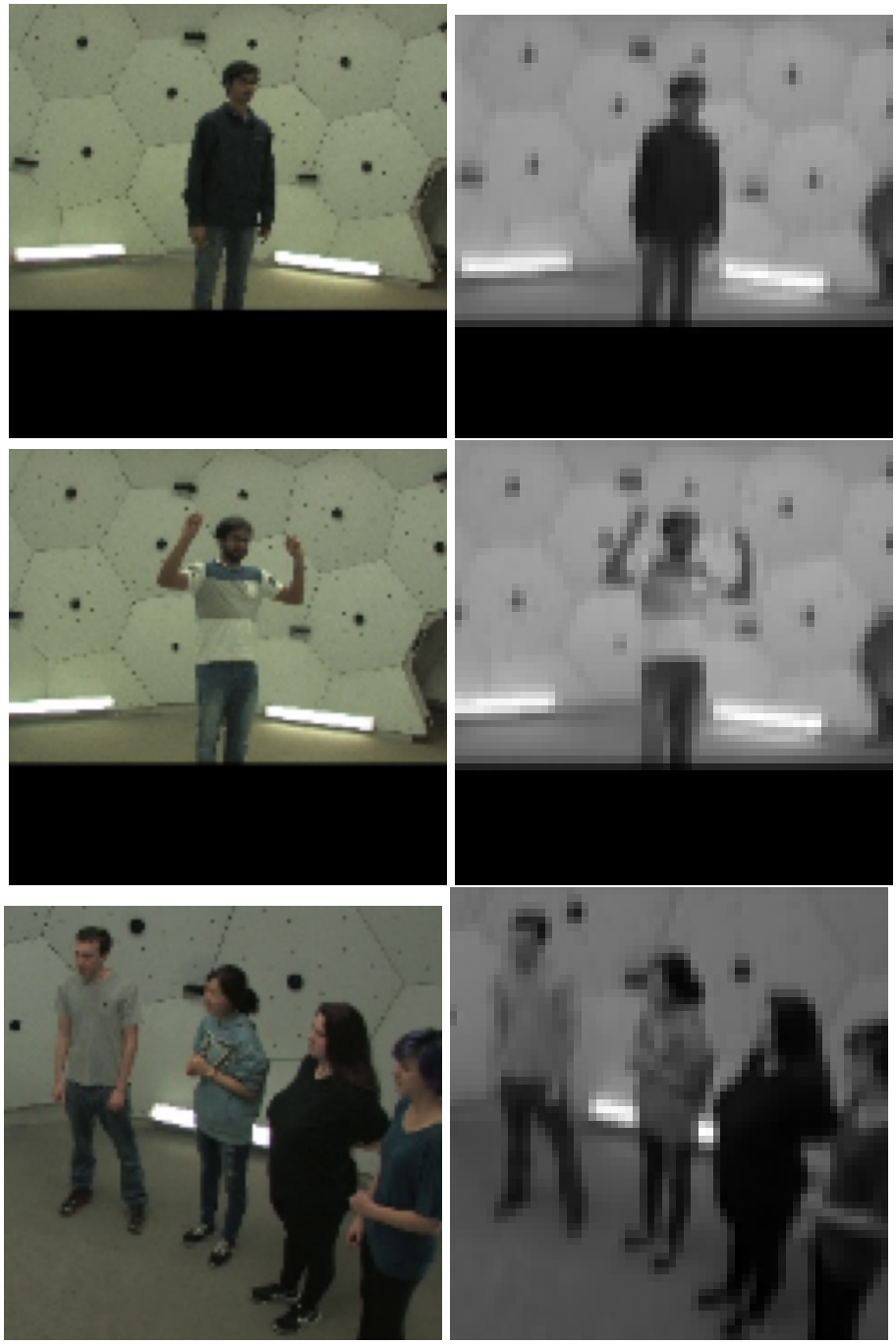


Fig. A14: Comparison histogram inversion

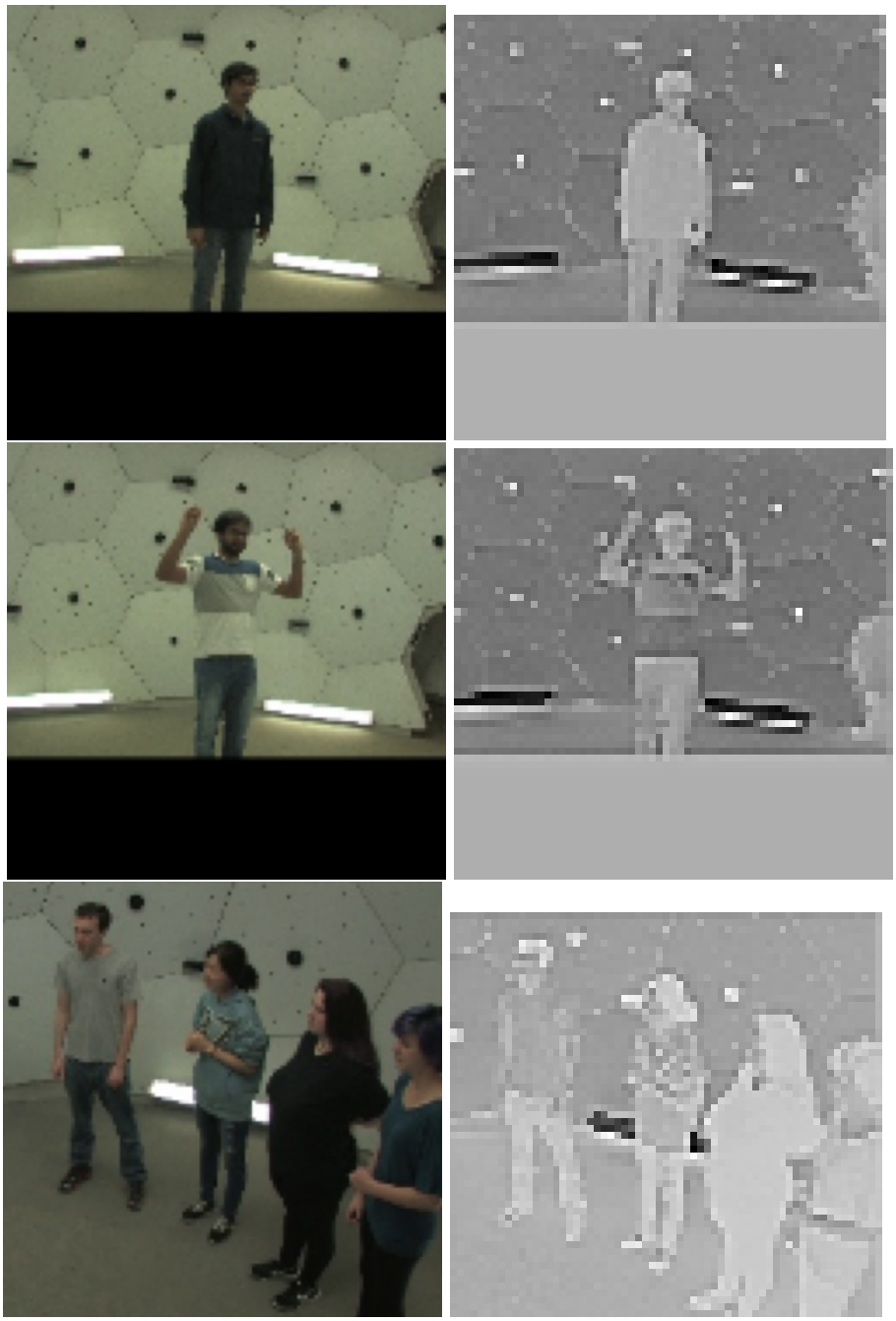


Fig. A15: Intermediate representations of SimCLR ResNet50 CNN

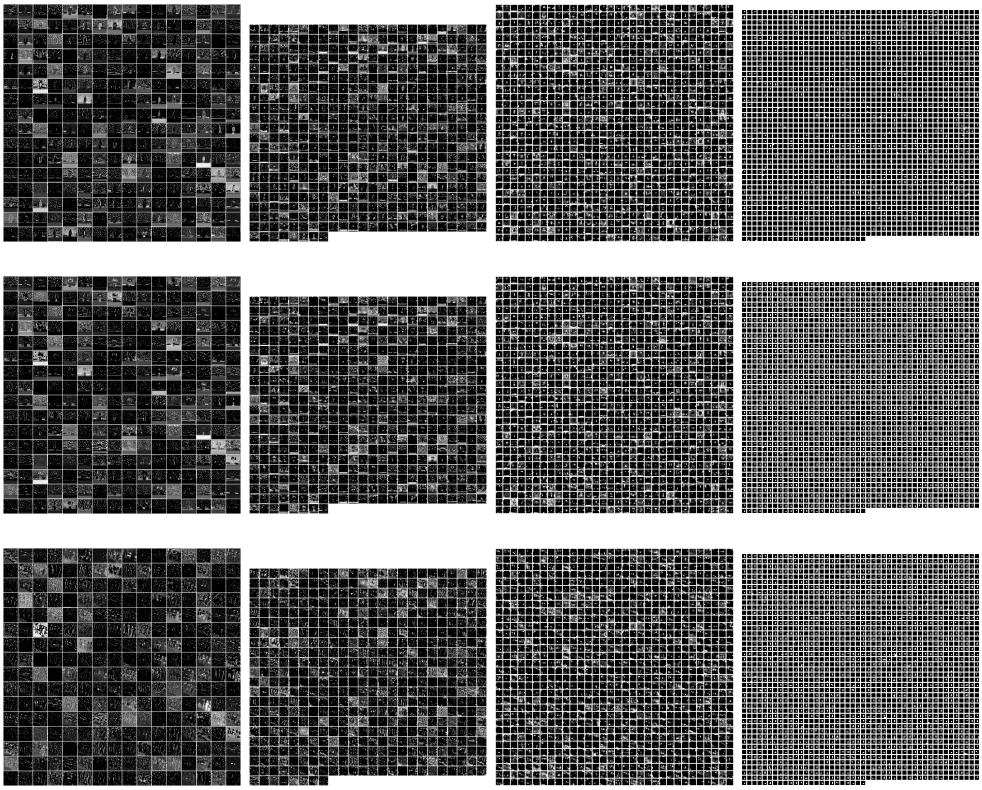
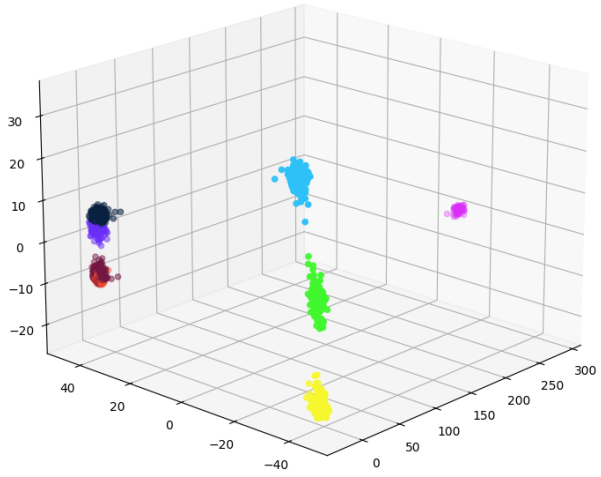


Fig. A16: LDA 3d projection for clusters obtained using the pre-trained SimCLR model

(a) Encoder features



(b) Projection head features

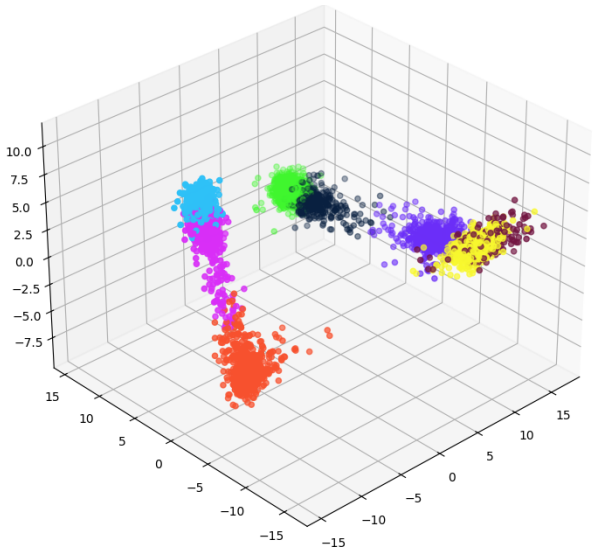
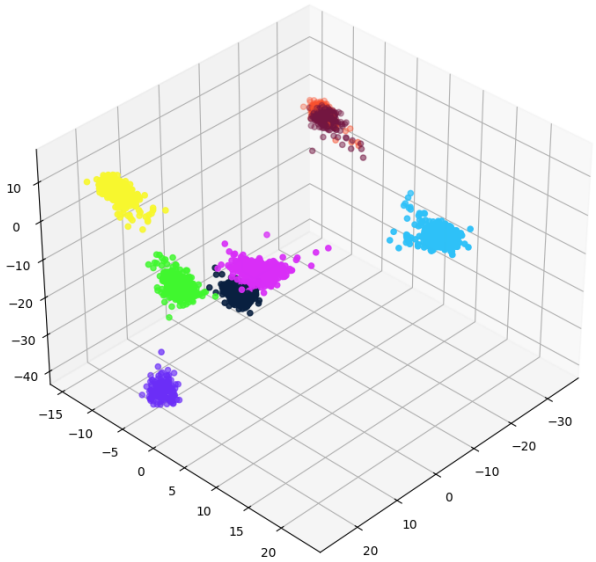


Fig. A17: LDA 3d projection for clusters obtained using our SimCLR model

(a) Encoder features



(b) Projection head features

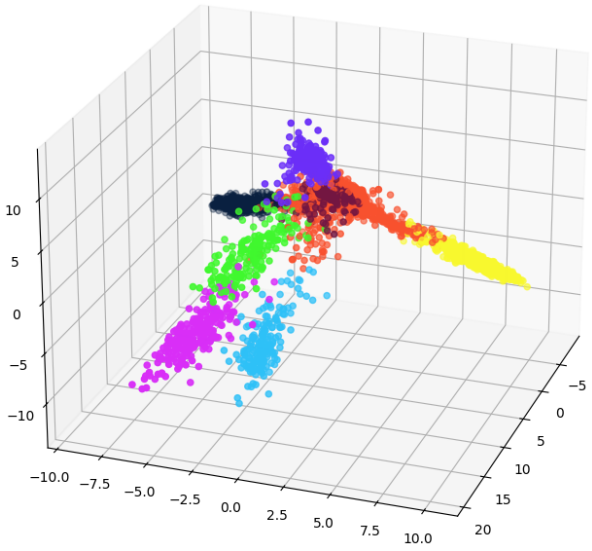


Fig. A18: results 3d pose estimation

(a) new SimCLR results

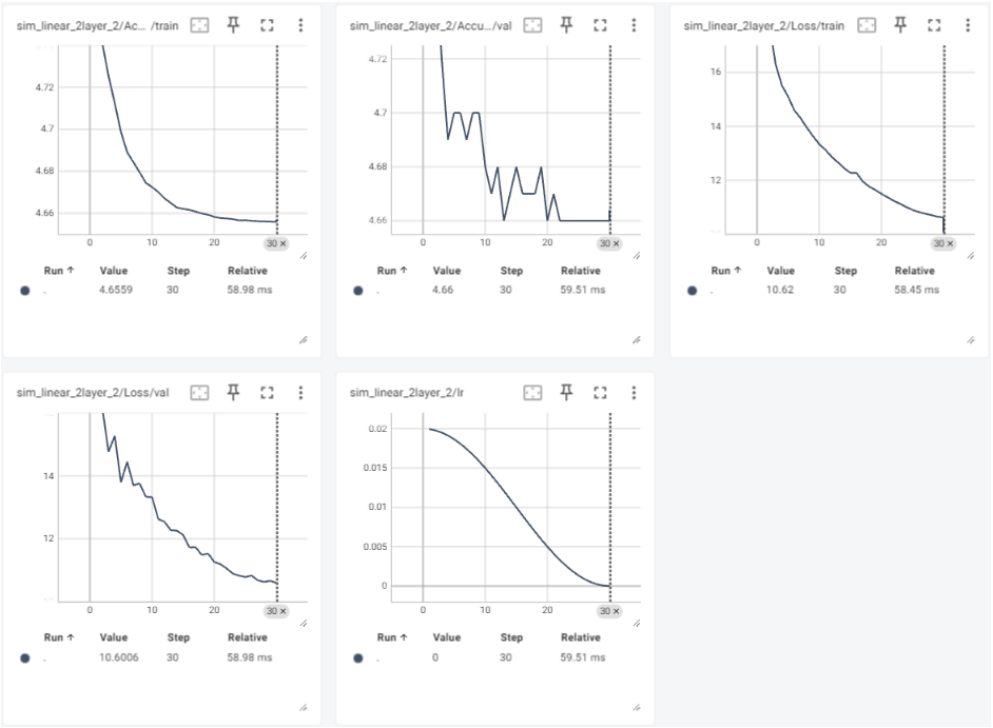


Fig. A19: new SimCLR predictions