1 Finetuning

As we now have finetuned our models, we need to finetune the models, such that they are specialized to yield optimial results on the ClimbAlong dataset. The following section describes the finetuning of these models. This includes the preprocessing of the data, the configuration details we use, as well as the obtained results.

In the finetuning stage we will be using the already developed pose-estiamtor to train our temporal-inclusive models. However, we will be freezing the pose-estimator, such that the weights of the model will not change during the training and we will thus only train our temporal-inclusive models. We do this for the following three reasons: (1) the training of the models will be quicker, as we just need to train the temporal-inclusive models and not the already developed pose-estimator, and (2) we get an greater understanding of the effects of our models when combined with the pose-estimator, as we can clearly see how big of a difference it makes by adding our temporal-inclusive models.

1.1 Data Preprocessing

For the ClimbAlong dataset we perform only minor preprocessing. First, the preprocessing of each video is done by having the already developed pose-estimator process the video, such that we have the output heatmaps of the pose-estimator, containing all of the pose-estimations of each video. Next, we preprocess the heatmaps by setting all negative values to 0 and normalizing each heatmap, such that each heatmap sums up to the fixed value $c = 255$ that we used when preprocessing the BRACE and Penn Action datasets, essentially making the heatmaps more similar to the preprocessed heatmaps of BRACE and Penn Action. These heatmaps will then be used as the input for our models.

For the groundtruth heatmaps we create twenty five heatmaps of each frame, similarly to how we did it for the BRACE and Penn Action datasets, however, in this case we use the predicted bounding-box of the pose-estimator as our bounding-box. In cases where the groundtruth keypoint is placed outside of the bounding-box, we place the groundtruth keypoint at the closest border of the bounding-box.

1.2 Training Details

Data Configuration Generally, we follow a similar approach to how we did in the pretraining stage. We again use a window-size of $k = 5$ frames, resulting in a total of $9,419$ windows. Also here are we using $c = 255$ as a representation of the ground turht placement of each keypoint. We also split the dataset into a non-overlapping and non-repeating training, validation and test set, consisting of 60%, 20% and 20% of the data, respectively. However, we note that one incorrect frame can have a huge impact on the evaluation results, as this frame is used five times during evaluation, and with the small dataset these five samples can have a huge impact. For that reason, for the validation and test set we make sure that no of the windows of the same set are overlapping.

Experiments As the finetuning dataset is so small, the fitting of the models is very quick, making us fit all of the 26 developed models from the pretraining stage. For each model we pick the epoch from the pretraining stage, that yielded the highest validation accuracy and use that for finetuning.

Training Configuration The optimization parameters are very similar to the ones from the pretraining stage. We again use the ADAM optimzer with a batch size of 16 and the MSE loss-function. During training, we again keep track of the lowest reached validation loss of an epoch and use learning-rate reduction and early-stopping in a similar manner to how we did in the pretraining stage. However, unlike the pretraining stage, we here use a smaller initial learning rate of 10^{-4}, as the weights only need to be fineadjusted, making us believe that greater learning rate would skew the weights too much.

1.3 Training and Validation Results

We have in Figure 1 illustrated the evaluation of the training loss, validation loss, and validation PCK@0.05 accuracy of the various models during the finetuning.

If we compare the models against the identity function we clearly see, how all models converge towards beating the identity function, indicating the positive effects of incorporating temporal information into pose estimation.

Generally, the 3-dimensional convolutional layer seems to be yielding the greatest results, as these models tend to achieve the highest validation PCK@0.05 accuracy. However, some DeciWatch runs actually starts off with an even higher validation PCK@0.05 accuracy, which then decreases to a much lower value quickly. The two architectures that are based on a bidirectional convolutional LSTM tend to yield some decent results, however, their training tend to plateau rather early, hence why they also terminate very quickly.

Further, the shifting-scalar seems to only have a minor effect on the models during the finetuning, as all six runs of each model tend converge towards the same result. This however does not hold for DeciWatch, as the DeciWatch models with shifting-scalar $s = 1$ seems to be having a lot of difficulty fitting compared to all of the other models, however, they do still yield some very accurate results.

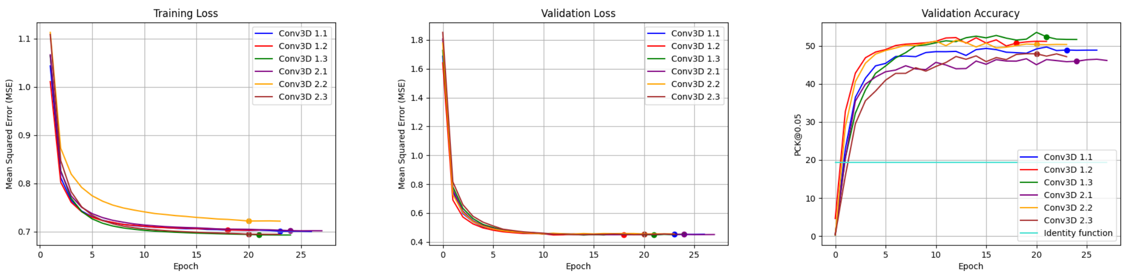
1.4 Test Results

Accuracy metric	PCK@0.05			PCK@0.1			PCK@0.2		
Mean threshold distance (px)*	0.80			1.60			3.21		
Experiment	1.1	1.2	1.3	1.1	1.2	1.3	1.1	1.2	1.3
Identity function	19.4	19.4	19.4	66.1	66.1	66.1	85.2	85.2	85.2
Conv3D	49.7	52.3	53.1	**95.7**	**95.7**	**95.8**	99.2	99.3	99.3
DeciWatch	**76.9**	**76.7**	**68.7**	94.3	94.4	86.9	99.2	99.2	96.2
bi-ConvLSTM - sum.	37.8	34.9	39.0	91.8	92.1	92.2	99.4	**99.7**	99.2
bi-ConvLSTM - concat.	35.9	39.0	38.5	93.1	93.6	92.6	**99.8**	**99.7**	**99.7**

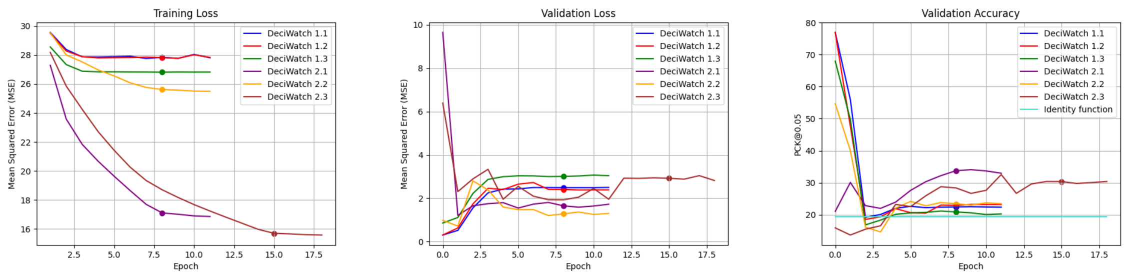
Table 1: Testing accuracies of the various developed models for shifting-scalar $k = 1$. All the accuracies are in percentage. *: The mean maximum distance between the predicted keypoint and corresponding groundtruth keypoint for the prediction to count as being correct, measured in the heatmap coordinate system.

We have in Table 1 and Table 2 illustrated the testing results of the epoch of each model that yielded the highest validation PCK@0.05 accuracy.

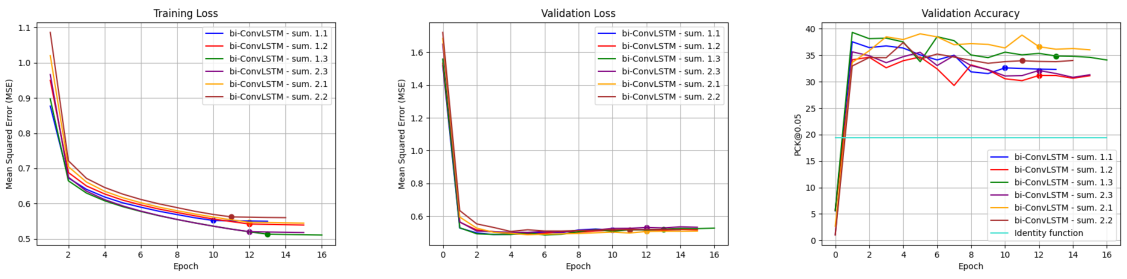
By comparing the two tables against each other we see, that the shifting-scalar tend to have a big effect on the results of DeciWatch. This is very similar to what we experienced in section 1.3, where we saw that the DeciWatch models with shifting-scalar $s = 1$ had a lot of difficulty



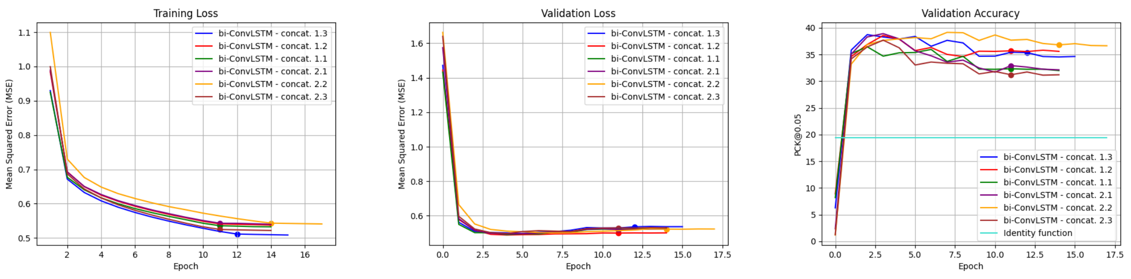
(a) Finetuning results of the 3-dimensional convolutional layer.



(b) Finetuning results of DeciWatch.



(c) Finetuning results of the bidirectional convolutional LSTM with summing.



(d) Finetuning results of the bidirectional convolutional LSTM with concatenation.

Figure 1: Evolution of the training loss, validation loss and validation PCK@0.05 accuracy of the 24 models during training, as well as the validation PCK@0.05 accuracy of the identity function of the two datasets. The dots indicates a reduction of learning-rate. First row: 3-dimensional convolutional layer. Second row: DeciWatch. Third row: Bidirectional Convolutional LSTM with summing. Fourth row: Bidirectional Convolutional LSTM with concatenation.

Accuracy metric	PCK@0.05			PCK@0.1			PCK@0.2		
Mean threshold distance (px)*	0.80			1.60			3.21		
Experiment	2.1	2.2	2.3	2.1	2.2	2.3	2.1	2.2	2.3
Identity function	19.4	19.4	19.4	66.1	66.1	66.1	85.2	85.2	85.2
Conv3D	**46.5**	**51.6**	**47.3**	**95.5**	**95.5**	**95.8**	99.2	99.3	99.2
DeciWatch	34.4	48.9	32.3	79.4	69.0	71.6	95.8	86.3	92.3
bi-ConvLSTM - sum.	38.8	37.4	35.9	92.7	92.1	91.2	99.4	**99.5**	99.3
bi-ConvLSTM - concat.	39.2	39.5	37.1	92.5	92.9	92.6	**99.6**	99.3	**99.6**

Table 2: Testing accuracies of the various developed models for shifting-scalar $k = 2$. All the accuracies are in percentage. *: The mean maximum distance between the predicted keypoint and corresponding groundtruth keypoint for the prediction to count as being correct, measured in the heatmap coordinate system.

fitting. For the remaining models we do see a performance difference between the two shifint-scalars, however, the performance difference is not as big as it is for DeciWatch. Generally however, the models do perform best with the noise-scalar $s = 1$.

Similarly to the pretraining stage, we also see here how the 3-dimensional convolutional layer from experiment 2 performs better than the 3-dimensional convolutional layer from experiment 1, however, this performance difference is now smaller than it was in the pretraining stage.

Further, for experiment 3 DeciWatch is not delivering great results, similarly to its results in the pretraining stage. For the other architectures, there do not seem to be any consistent pattern, as some of them experiment 3 yields the best results, whereas it for others yield the worst results.

Like in the pretraining stage, there does not seem to be any major performance differences between the two architectures that are based on the bidirectional convolutional LSTM, making us further believe that our concern about the missing opportunity of the model to prioritize a processing direction is not that important.

	Conv3D			DeciWatch			bi-ConvLSTM sum.			bi-ConvLSTM concat.			Total
Experiment	1.1	1.2	1.3	1.1	1.2	1.3	1.1	1.2	1.3	1.1	1.2	1.3	
Nose	100	100	100	99.8	99.8	97.5	100	100	99.9	100	99.7	99.9	
Ear	100	100	100	99.8	99.8	99.8	97.7	99.9	100	100	100	99.9	
Shoulder	99.9	100	99.9	99.8	99.8	97.4	100	100	99.9	100	100	100	
Elbow	99.9	99.9	99.9	99.4	99.4	96.9	100	100	100	100	99.9	100	
Wrist	99.8	99.9	99.9	99.1	99.2	96.2	100	99.9	99.8	100	99.9	100	
Pinky	93.4	93.1	94.4	98.3	98.4	94.4	97.2	98.8	97.0	98.0	99.0	98.6	
Index finger	99.0	98.8	98.8	98.2	98.3	93.8	99.5	98.7	97.0	99.6	99.4	99.4	
Thumb	98.9	98.8	98.9	98.2	98.3	95.0	96.8	99.6	97.8	99.7	98.6	99.6	
Hip	99.9	100	100	99.7	99.7	97.6	100	100	100	100	100	100	
Knee	100	100	99.9	99.7	99.6	97.3	100	100	100	100	100	100	
Ankle	100	100	100	99.5	99.5	96.8	100	100	99.9	100	100	99.9	
Heel	100	100	100	99.3	99.3	96.1	99.3	99.9	99.9	99.9	100	99.8	
Toes	99.9	100	100	99.0	99.1	95.2	99.6	99.8	99.4	99.8	100	99.8	

Table 3: Keypoint-specific testing PCK@0.2-accuracies of the various models for shiting-scalar $k = 1$. All the accuracies are in percentage.

	Conv3D			DeciWatch			bi-ConvLSTM sum.			bi-ConvLSTM concat.			Total
Experiment	2.1	2.2	2.3	2.1	2.2	2.3	2.1	2.2	2.3	2.1	2.2	2.3	
Nose	100	100	100	99.0	99.8	97.2	100	99.9	99.7	99.7	99.9	100	
Ear	100	99.8	100	99.2	86.3	90.5	99.8	99.8	100	99.9	99.9	99.9	
Shoulder	99.9	99.7	99.9	43.8	99.6	97.1	99.9	99.8	100	99.9	100	100	
Elbow	99.8	99.9	99.9	93.3	65.8	91.6	100	99.5	100	100	100	100	
Wrist	99.8	100	99.9	94.1	98.4	93.9	99.8	99.7	99.8	99.8	100	99.7	
Pinky	93.1	93.7	93.9	91.9	98.3	87.3	97.7	98.0	97.9	99.1	96.0	98.2	
Index finger	98.9	99.0	98.8	98.2	97.9	92.3	99.4	99.1	99.2	99.6	98.0	97.5	
Thumb	98.6	98.6	98.6	96.5	98.2	92.8	95.7	96.6	98.6	97.5	98.3	99.6	
Hip	100	99.9	100	96.7	38.3	91.7	99.9	99.8	99.9	99.8	100	99.9	
Knee	100	100	99.9	95.1	99.5	91.8	100	99.9	99.9	99.9	99.9	100	
Ankle	100	99.9	100	94.1	99.5	92.7	100	100	100	100	100	100	
Heel	100	100	100	96.1	99.3	89.5	99.9	99.9	99.8	99.9	99.6	99.9	
Foot	99.9	100	100	97.2	99.1	93.7	99.9	99.4	98.4	99.8	99.6	99.8	

Table 4: Keypoint-specific testing PCK@0.2-accuracies of the various models for shiting-scalar $k = 2$. All the accuracies are in percentage.

We have in Table 3 and Table 4 illustrated the keypoint specific testing accuracies of the models. By comparing these tables to the equivalent tables from section **??** we see, that most difficult keypoints are no longer the wrists and ankles, but are instead the pinkies, the index fingers and the thumbs, on which the models generally performs much worse than on the remaining keypoints.

1.5 Technical Details

All models were trained and evaluated using a 8GB NVIDIA GTX 1070 and an Intel Core i7-4790K @ 4.00GHz. All models were implemented in Python version 3.9.9 using PyTorch 2.0.0. The 3-dimensional convolutional layer took about 1.5 minutes per epoch, DeciWatch about 2 minutes per epoch, and the two bidirectional convolutional LSTMs took about 2.5 minutes per epoch each.