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 keypoint detector 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 as (1) the training of the models will be quicker, as we just need to train the temporal-inclusive models and not the keypoint detector, 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.

In section 1.1 we cover the preprocessing of the finetune dataset. Then, in section 1.2 we cover the used configuration for the finetuning. Then, in section 1.3 and section 1.4 we cover the training and validation results, as well as the testing results, respectively. Lastly, in section 1.5 we cover the technical details of the finetuning.

1.1 Data Preprocessing

For the ClimbAlong dataset we perform only minor preprocessing. First, the preprocessing of each video is done by having the keypoint detector 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, for the data configuration 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 $10,173$ windows. Also here are we using $c = 255$ as a representation of the ground truth placement of each keypoint.

As this dataset is much smaller the the dataset used during the pretraining-stage, we can much more easily introduce some evaluation-bias, hence why we also take much more careful steps. Thus, the splitting of the dataset is different than how we performed it in the pretraining-stage. First, we extract the longest video, such that none of its windows are on the dataset, which we will be used to unsure an unbias evaluation of our models. Next, we use the first 60% of the frame-windows and use those as the training dataset. For the remaining 40% of the frame-windows, we make sure that they have no overlapping frames with training dataset - if they do, the whole window is moved into the training dataset. As we still have to split the

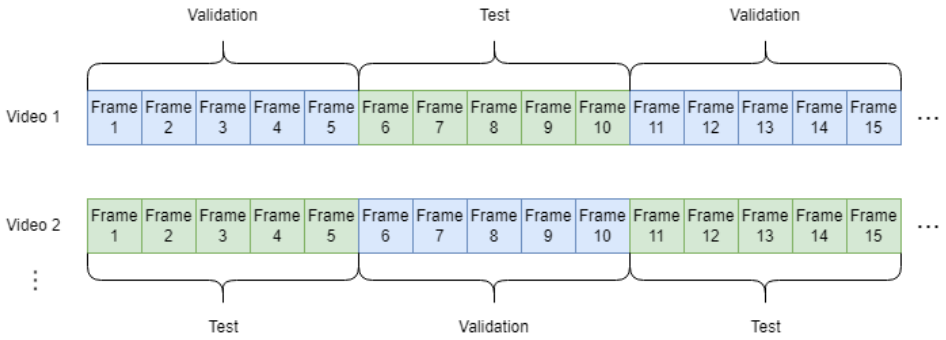


Figure 1: Illustration of how the ClimbAlong data has been split into a validation dataset and a test dataset.

remaining 40% of the frame-windows into a validation dataset and a test dataset, we have to make sure, that (1) both datasets use as many video sequences as possible, which is especially important in this case due to the small dataset size, (2) that none of the windows are overlapping among the two datasets, and (3) both datasets sample the windows uniformly distributed throughout the video sequences, as there can be some sepcial movements in certain parts of the video sequences. All of these constraints will make sure that we minimize the evaluation-bias of our models. The splitting of the remaining 40% of the window into two datasets is then done by sorting the non-overlapping windows by their video and by their time of occurence in their corresponding video. Then, if the ith window-frame is divisible by two, it is moved into the validation dataset, otherwise it is moved into the test dataset. By doing so we meet all three of our conditions. The overall idea has been illustrated in Figure 1.

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

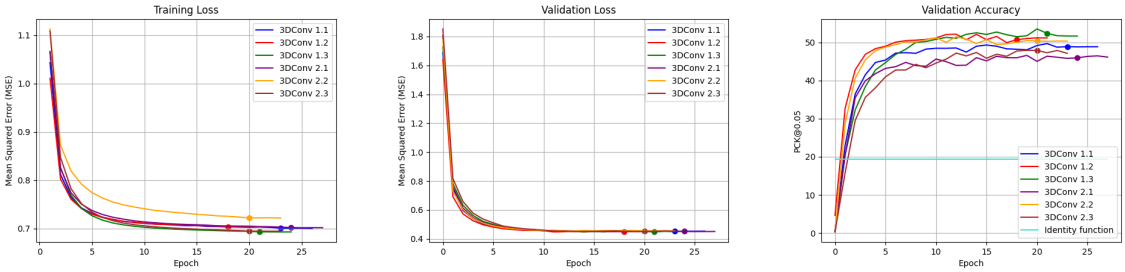
Training Configuration The optimization parameters are very similar to the ones from the pretraining stage. We again use the MSE loss-function, a batch size of 16, and the ADAM optimizer with an initial learning rate of 10^{-3}, and $\rho_1 = 0.9$ and $\rho_2 = 0.999$ as these three values were suggested by [1]. 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 iniitial learning rate of 10^{-4}, as the weights only need to be fineadjusted, making us believe that a greater learning rate would skew the weights too much.

1.3 Training and Validation Results

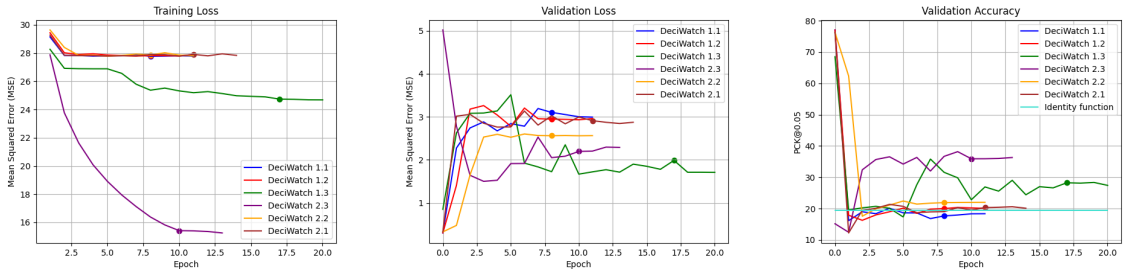
We have in Figure 2 illustrated the evaluation of the training loss, validation loss, and validation PCK@0.05 accuracy of the various models during the finetuning.

If we compare the models against the identity function we clearly see, how all models at some point beats the identity function, indicating the positive effects of incorporating temporal information into pose estimation.

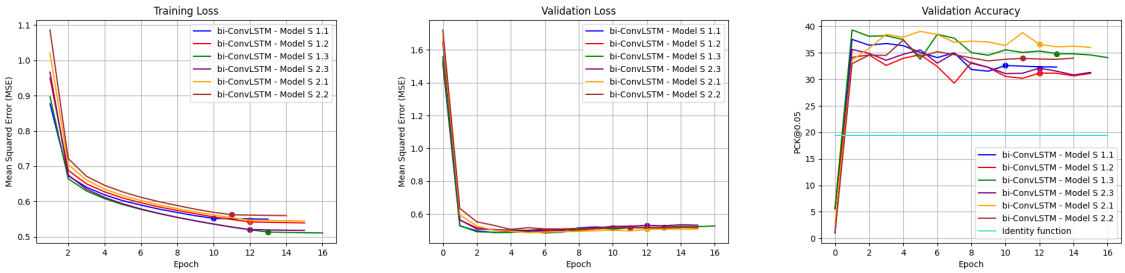
Generally, 3DConv seems to be converging towards the greatest results, as these models tend to converge towards the highest validation PCK@0.05 accuracy. However, some DeciWatch



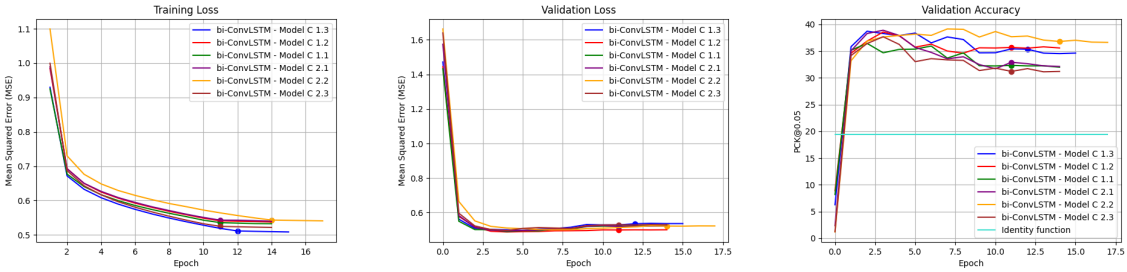
(a) Finetuning results of 3DConv.



(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 2: 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: 3DConv. Second row: DeciWatch. Third row: Bidirectional Convolutional LSTM with summing. Fourth row: Bidirectional Convolutional LSTM with concatenation.

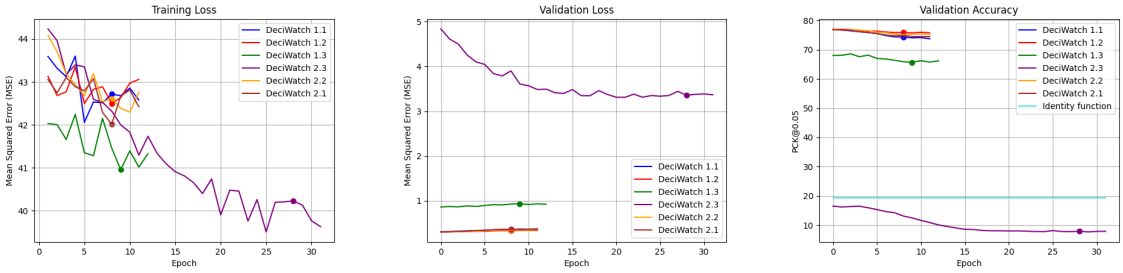


Figure 3: Finetuning results of DeciWatch with regularization techniques.

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. Generally however, both DeciWatch and the two architectures based on a bidirectional convolutional LSTM do seem to overfit. For DeciWatch this is mostly immediately, whereas for the bidirectional convolutional LSTMs this tend to happen a few epochs later.

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.

1.3.1 Additional Experiment Results

As seen in Figure 2, most of the DeciWatch methods would immediately overfit, as the validation loss increased while the training loss decreased, leading to the validation accuracy decreasing overtime. We performed further experiments, where we implemented various regularization techniques in hopes of this would lower the likelihood of the models overfitting. This was done by (1) freezing the whole network except for the very last fully-connected layer, which we reinitialized randomly, (2) added variance to the data by rotating each window throughout the training with a random degree in the range $[-45, 45]$, and (3) by applying weight decay with $\lambda = 0.001$.

By incorporating these regualrization techniques we obtain the results visualized in Figure 3. While the training loss doss generally decrease much slower than previously, the validation accuracy is still decreasing, making the best setting of the models without these regualrization methods perform either at the same level or an even better level than the models with regualrization.

1.4 Test Results

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 only have a minor effect on the results of the models. Generally however, the models do perform best with the noise-scalar $s = 1$.

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

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
3DConv	49.7	52.3	53.1	**95.7**	**95.7**	**95.8**	99.2	99.3	99.3
DeciWatch	**76.6**	**76.7**	**68.1**	94.4	94.3	87.3	99.2	99.2	96.1
bi-ConvLSTM - Model S	37.8	34.9	39.0	91.8	92.1	92.2	99.4	**99.7**	99.2
bi-ConvLSTM - Model C	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.

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
3DConv	46.5	51.6	**47.3**	**95.5**	**95.5**	**95.8**	99.2	99.3	99.2
DeciWatch	**76.0**	**75.9**	36.8	94.2	94.2	74.9	99.2	99.2	92.8
bi-ConvLSTM - Model S	38.8	37.4	35.9	92.7	92.1	91.2	99.4	**99.5**	99.3
bi-ConvLSTM - Model C	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.

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 biConv-LSTM Model S to prioritize a processing direction is not that important.

Accuracy metric	PCK@0.05			PCK@0.1			PCK@0.2		
Mean threshold distance*	0.87			1.77			3.55		
Experiment	1.1	1.2	1.3	1.1	1.2	1.3	1.1	1.2	1.3
Identity function	21.2	21.2	21.2	65.5	65.5	65.5	84.7	84.7	84.7
3DConv	58.4	61.4	61.7	**98.7**	**98.9**	**99.0**	**99.6**	**99.8**	**99.7**
DeciWatch	**82.6**	**82.4**	**74.6**	96.2	96.1	92.3	99.1	99.1	97.4
bi-ConvLSTM - Model S	45.7	45.0	47.6	97.3	96.9	97.0	**99.6**	**99.6**	99.1
bi-ConvLSTM - Model C	44.5	46.1	48.5	97.4	97.9	97.9	99.6	99.5	99.6

Table 3: Testing accuracies of the various developed models for shifting-scalar $k = 1$ on the additional test video. 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, using the units of the heatmap coordinates.

Accuracy metric	PCK@0.05			PCK@0.1			PCK@0.2		
Mean threshold distance*	0.87			1.77			3.55		
Experiment	2.1	2.2	2.3	2.1	2.2	2.3	2.1	2.2	2.3
Identity function	21.2	21.2	21.2	65.5	65.5	65.5	84.7	84.7	84.7
3DConv	56.2	60.0	**56.6**	**98.9**	**98.8**	**98.8**	**99.7**	**99.7**	**99.7**
DeciWatch	**81.6**	**81.8**	37.5	96.0	96.0	73.3	99.1	99.1	90.7
bi-ConvLSTM - Model S	44.8	46.2	45.0	96.9	95.9	97.1	99.5	99.6	99.5
bi-ConvLSTM - Model C	45.9	47.9	46.7	96.7	97.1	98.1	99.6	99.4	99.6

Table 4: Testing accuracies of the various developed models for shifting-scalar $k = 2$ on the additional test video. 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, using the units of the heatmap coordinates.

One might argue, that our way of splitting the finetuning dataset into a validation and test dataset might introduce some bias, as the two datasets have overlapping videos and thus videos of the same person that might have some unique movement. To further minimize the likelihood of a biased evaluation of the models, we have in Table 3 and Table 4 tested the models on the longest video of the dataset, which was not used at all in any of the three data-subsets and consists of 754 overlapping windows. By comparing these tables against Table 1 and 2 we can clearly see, that the models are performing similarly, if not actually better, than how they did previously, hence why we would argue, that there are no indication of the first test evaluation being biased. However, we have to note, that the evaluation in Table 3 and Table 4 do contain some bias, as the evaluation is of a single person who might have some easily predictable movements.
We have in Table 5 and Table 6 illustrated the keypoint specific testing accuracies of the models. By comparing these tables to the equivalent tables from section **??** we see, that most difficult

	3DConv			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	99.8	100	100	99.9	100	99.7	99.9	99.9
Ear	100	100	100	99.8	99.8	99.8	97.7	99.9	100	100	100	99.9	99.7
Shoulder	99.9	100	99.9	99.8	99.8	99.8	100	100	99.9	100	100	100	99.9
Elbow	99.9	99.9	99.9	99.4	99.4	99.4	100	100	100	100	99.9	100	99.8
Wrist	99.8	99.9	99.9	99.1	99.2	99.1	100	99.9	99.8	100	99.9	100	99.7
Pinky	93.4	93.1	94.4	98.3	98.4	98.3	97.2	98.8	97.0	98.0	99.0	98.6	97.0
Index finger	99.0	98.8	98.8	98.2	98.2	98.2	99.5	98.7	97.0	99.6	99.4	99.4	98.7
Thumb	98.9	98.8	98.9	98.3	98.3	98.3	96.8	99.6	97.8	99.7	98.6	99.6	98.6
Hip	99.9	100	100	99.7	99.7	99.7	100	100	100	100	100	100	99.9
Knee	100	100	99.9	99.7	99.7	99.7	100	100	100	100	100	100	99.9
Ankle	100	100	100	99.5	99.5	99.5	100	100	99.9	100	100	99.9	99.9
Heel	100	100	100	99.2	99.2	99.2	99.3	99.9	99.9	99.9	100	99.8	99.7
Toes	99.9	100	100	99.1	99.0	99.1	99.6	99.8	99.4	99.8	100	99.8	99.6
Total	99.2	99.3	99.3	99.2	99.2	96.1	99.4	99.7	99.2	99.8	99.7	99.7	

Table 5: Keypoint-specific testing PCK@0.2-accuracies of the various models for shiting-scalar $k = 1$. All the accuracies are in percentage.

	3DConv			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	99.8	99.8	97.2	100	99.9	99.7	99.7	99.9	100	99.7
Ear	100	99.8	100	99.7	99.7	97.1	99.8	99.8	100	99.9	99.9	99.9	99.6
Shoulder	99.9	99.7	99.9	99.8	99.9	95.8	99.9	99.8	100	99.9	100	100	99.6
Elbow	99.8	99.9	99.9	99.5	99.4	90.8	100	99.5	100	100	100	100	99.1
Wrist	99.8	100	99.9	99.1	99.1	94.4	99.8	99.7	99.8	99.8	100	99.7	99.3
Pinky	93.1	93.7	93.9	98.4	98.4	86.5	97.7	98.0	97.9	99.1	96.0	98.2	95.9
Index finger	98.9	99.0	98.8	98.2	98.2	88.9	99.4	99.1	99.2	99.6	98.0	97.5	97.9
Thumb	98.6	98.6	98.6	98.3	98.2	90.1	95.7	96.6	98.6	97.5	98.3	99.6	97.4
Hip	100	99.9	100	99.7	99.8	95.1	99.9	99.8	99.9	99.8	100	99.9	99.5
Knee	100	100	99.9	99.6	99.7	95.1	100	99.9	99.9	99.9	99.9	100	99.5
Ankle	100	99.9	100	99.5	99.5	93.9	100	100	100	100	100	100	99.4
Heel	100	100	100	99.3	99.3	91.0	99.9	99.9	99.8	99.9	99.6	99.9	99.1
Toes	99.9	100	100	99.0	99.1	91.9	99.9	99.4	98.4	99.8	99.6	99.8	98.9
Total	99.2	99.3	99.2	99.2	99.2	92.8	99.4	99.5	99.3	99.6	99.3	99.6	

Table 6: Keypoint-specific testing PCK@0.2-accuracies of the various models for shiting-scalar $k = 2$. All the accuracies are in percentage.

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.

For some illustrations of the predictions of the best setup of each model based on Table **??** and Table **??**, see Figure **??** in the appendix.

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. 3DConv 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.