

Scriboora: rethinking human pose forecasting

Daniel Bermuth, Alexander Poeppel, Wolfgang Reif

Angaben zur Veröffentlichung / Publication details:

Bermuth, Daniel, Alexander Poeppel, and Wolfgang Reif. 2026. "Scriboora: rethinking human pose forecasting." arXiv. arXiv. <https://doi.org/10.48550/arXiv.2511.15565>.

Nutzungsbedingungen / Terms of use:

CC BY-SA 4.0

Dieses Dokument wird unter folgenden Bedingungen zur Verfügung gestellt: / This document is made available under these conditions:
CC-BY-SA 4.0: Creative Commons: Namensnennung - Weitergabe unter gleichen Bedingungen
Weitere Informationen finden Sie unter: / For more information see:
<https://creativecommons.org/licenses/by-sa/4.0/deed.de>



Scriboora: Rethinking Human Pose Forecasting

Daniel Bermuth
University of Augsburg, Germany
daniel.bermuth@uni-a.de

Alexander Poeppl
University of Augsburg
poeppl@isse.de

Wolfgang Reif
University of Augsburg
reif@isse.de

Abstract

Human pose forecasting predicts future human poses from past observations for applications including action recognition, sports analysis, and human-robot interaction. However, progress in the field is difficult to assess because reported results often rely on heterogeneous preprocessing, inconsistent metric implementations, and partially released artifacts. This paper re-examines human pose forecasting as an evaluation problem by auditing a wide range of recent forecasting methods, identifying several reproducibility issues, and introducing a unified training and evaluation pipeline for absolute pose forecasting. Under this unified protocol, mature cross-domain sequence models adapted from speech recognition provide strong baselines and obtain the lowest errors among the evaluated models. To study deployment-relevant noise, a paired clean/detector-noisy dataset variant is introduced using poses generated by a pose-estimation model. The experiments show that detector-generated poses cause substantial performance degradation compared with clean motion-capture inputs, but that part of this degradation can be recovered by adapting the forecaster on detector-generated pose sequences.

1. Introduction

Human pose forecasting represents observed motion as sequences of body-joint coordinates and predicts future joint trajectories from past observations (see Figure 1). Such forecasts can support applications such as predicting pedestrian movements or replanning robot motions in collaborative settings.

Although many methods report rapid progress, the state of the field remains difficult to assess. Results are often produced with heterogeneous preprocessing, inconsistent metric implementations, partially released artifacts, and sometimes evaluation errors. Comparisons across papers are therefore unreliable, and claimed improvements can vanish once consistent protocols are applied. Prior evaluations also tend to compare only against specialized pose-forecasting baselines, leaving unclear whether mature sequence models from neigh-

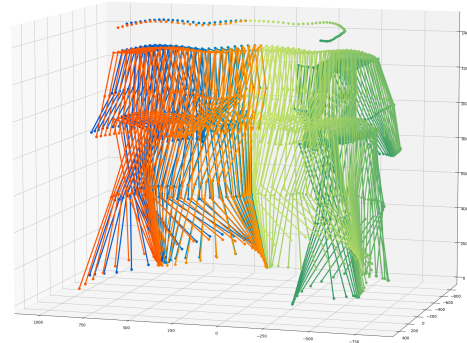


Figure 1. Example of an *absolute* pose forecast of a walking person. The green skeletons visualize the input sequence of the prediction model, the red ones the predicted future poses, and the blue ones the ground-truth labels.

boring domains already provide strong baselines. At the same time, evaluations rarely reflect deployment conditions. In practical systems, input poses often come from marker-less human pose estimators, since outfitting pedestrians or factory workers with tracking suits is usually infeasible. The errors of such estimators differ from synthetic noise used in prior studies and remain underexplored for pose forecasting.

This paper addresses these issues with a practice-oriented evaluation study of absolute pose forecasting, making three contributions: (1) a broad reproducibility audit of recent forecasting methods; (2) a unified absolute-pose evaluation showing that prior rankings do not necessarily transfer and that mature cross-domain sequence baselines obtain lower errors than the evaluated specialized models under the same protocol; (3) a paired clean/detector-noisy evaluation for measuring pose-estimator-induced degradation, together with pseudo-label adaptation to recover part of the lost performance. Taken together, these contributions are intended to provide a repeatable and application-targeted benchmark protocol that enables future research to better compare and evaluate pose-forecasting methods.

To support subsequent work in these directions, model and preprocessing code, evaluation tools, and proposed-model checkpoints are released at: <https://gitlab.com/Percipote/>

2. Related Work

Most recent pose forecasting approaches have specialized into two categories: (1) *Relative* pose forecasting, which predicts a future pose relative to the last input pose, and therefore does not include global human motion trajectories (*HisReplItself* [38], *STSGCN* [31], *SPGSN* [20], *PGBIG* [21], *MotionMixer* [5], *siMLPe* [11], *EqMotion* [41], *DePOSit* [28], *AuxFormer* [40], *GMFnet* [30], *GCNext* [37], *DeformMLP* [13]). (2) *Absolute* pose forecasting, which predicts a future pose in absolute coordinates, and therefore also includes global human motion trajectories (*SoMoFormer* [34], *TBiFormer* [27], *JRTransformer* [42], *IAFormer* [39], *EMPMP* [43]). Appendix A provides additional architectural details.

The *mean per joint position error* (*MPJPE*, Equation 1) is the most commonly used metric to compare different approaches. It computes the distance between each predicted joint position and the ground-truth position. Afterward, those distances are averaged across all joints and persons to obtain a single score. When considering a forecast with multiple timesteps, the *MPJPE* score is only calculated for the prediction of a specific timestep. *MPJPE* is reported in millimeters throughout this paper.

$$MPJPE_t = \frac{\sum_{p=1}^{persons} \sum_{j=1}^{joints} \sqrt{\sum_{k=1}^3 (gt_{t,p,j,k} - pred_{t,p,j,k})^2}}{persons \cdot joints} \quad (1)$$

Depending on the two different categories, the commonly used datasets and evaluation protocols differ as well. For relative pose forecasting, the standard dataset is the *Human3.6M* [14] dataset, where actors are recorded by 4 cameras placed in the room corners while performing 15 different actions. Most relative approaches build upon the source-codes from [24, 25] and reuse their preprocessed datasets and evaluation metrics. The absolute pose forecasting approaches, on the other hand, are usually evaluated on the *3D Poses in the wild* (3DPW) dataset [35], which consists of 60 action sequences in a real-world setting, or the *CMU-MoCap* [12] dataset, in which the actors have been captured by twelve cameras in a $3m \times 8m$ room.

Most of those approaches have focused on using ground-truth pose labels from the dataset, and only a few approaches [6, 28] also added artificial errors. For several reasons, which are explained in more detail in Section 8, such artificial errors differ from the pose estimation errors. Therefore, the impact of such errors will be investigated later on, using paired clean and detector-generated pose sequences, which allow the detector-induced forecasting error to be measured against ground truth.

3. Reproducibility Audit

Before constructing a unified benchmark, the reported results of recent relative and absolute pose-forecasting methods were audited using the authors’ released artifacts where available. The goal of the audit is not only to check whether numbers can be rerun, but also to identify which evaluation assumptions make reported comparisons fragile.

For each method, the audit recorded whether source code, checkpoints, preprocessing scripts, and evaluation scripts were available. It then tested whether the reported numbers could be reproduced without modifying the released evaluation procedure (allowing differences of up to 1mm). When reproduction was unsuccessful, the dominant discrepancy source was categorized as a metric implementation issue, missing or partial artifacts, checkpoint-only reproducibility, training mismatch, unit or scale ambiguity, timestamp ambiguity, or environment failure. When a released evaluation procedure contained an identifiable metric bug, both the original reproduced behavior and the corrected result were recorded. Table 1 summarizes the observed failure modes, while detailed numerical comparisons and per-method notes are provided in Appendix B.

Method	Reproduction outcome	Main issue
MotionMixer [5]	corrected result lower	averaging/counting bug
STSGCN [31]	corrected result lower	MPJPE implementation bug
GMFnet [30]	checkpoint reproduced	training/checkpoint mismatch
SPGSN [20]	retrained result lower	unresolved training mismatch
HisReplItself [38]	reproduced by code	–
AuxFormer [40]	checkpoint reproduced	training/checkpoint mismatch
PGBIG [21]	reproduced by code	–
EqMotion [41]	checkpoint reproduced	training/checkpoint mismatch
GCNext [37]	reproduced by code	–
CSCNet [19]	no code to reproduce	missing artifact
DePOSit [28]	checkpoint reproduced	training mismatch unresolved
siMLPe [11]	reproduced by code	–
DeformMLP [13]	reproduced by code	–
IAFormer [39]	partially reproduced	missing H36M implementation
TRIPOD [1]	metric discrepancy	partial code, unintended metric behavior
MRT [36]	ambiguous	partial code, metric unit ambiguity
SoMoFormer [34]	ambiguous	partial code, metric unit ambiguity
TBiFormer [27]	ambiguous	partial code, metric unit ambiguity
T2P [15]	not evaluated	partial code, environment setup failure
EMPMP [43]	ambiguous	partial code, metric unit ambiguity
JRTransformer [42]	ambiguous	partial code, metric unit ambiguity

Table 1. Summary of reproducibility audit. The table reports the dominant issue observed for each method. “Checkpoint reproduced” means that the released checkpoint matched the reported result, but training from the released code did not reproduce the same performance. “Reproduced by code” means that the model was successfully trained and evaluated using the publicly released code. “Corrected result lower” means that an evaluation issue was identified and the corrected score was worse than the originally reported score. Detailed numerical comparisons and forensic notes are provided in Appendix B.

The audit showed that reproducibility problems are not limited to missing code. Several discrepancies arose from evaluation design choices that directly affect scientific conclusions, including metric implementations, preprocessing conventions, unit conversions, and forecast timestamps. In particular, correcting metric errors changed the apparent ranking of some methods, while incomplete preprocessing releases and ambiguous scale conventions made several absolute-pose benchmarks difficult to compare across papers. Overall, substantial reproducibility or evaluation issues were found across many of the examined methods. Reported improvements therefore cannot be interpreted reliably unless preprocessing, metric implementation, units, timestamps, and released artifacts are controlled. These findings motivate the unified protocol in Section 5 and highlight the need to release complete preprocessing and evaluation code together with trained checkpoints where possible.

4. Latency-Aware Metrics

Many uses of pose forecasting are latency-sensitive, but standard forecasting metrics ignore the time required to compute a forecast (measured by *forecasts per second (FPS)* at batch-size 1). For example, a model that requires more than one second to forecast the next second of motion has limited value in a real-time system, because the predicted interval has elapsed before the forecast is available. To measure how inference time affects the usefulness of a forecast once pose inputs are already available, two new latency-aware metrics are introduced. These metrics are intended to complement *MPJPE* when comparing forecasting models under a fixed upstream sensing pipeline.

The *forecast after delay error (FADE)*, Equation 2) focuses on average forecasting performance while accounting for inference delay. The idea is that if a prediction takes some amount of time to compute, the forecast must effectively reach that much further into the future. Because changing the output sequence length of every model would be impractical, *FADE* uses a simpler approximation that assumes linear error growth with time. Appendix D validates the linearity assumption in more detail.

$$FADE_t = MPJPE_t + MPJPE_t \cdot \frac{1000 \text{ ms}}{t} \cdot \frac{1}{FPS} \quad (2)$$

The *fast change estimate (FCE)*, Equation 3) is a conservative forecaster-side latency measure for abrupt motion changes. Using the human limb speed from *ISO 13855*, it estimates how far a limb could move during one model inference interval. After an unexpected motion change, the system relies on a prediction conditioned on outdated observations until a new forecast is computed. Lower *FCE* indicates faster reaction to sudden motion changes.

$$FCE = 2000 \text{ mm} \cdot \frac{1}{FPS} \quad (3)$$

In real applications, the image-to-pose system also introduces latency. This latency is excluded from *FADE* and *FCE* because it is deployment-specific and depends on the camera and hardware systems as well as on the pose-estimation method. For a fixed upstream pipeline, this latency is shared across forecasters and therefore does not affect their relative comparison. End-to-end deployment evaluations should add upstream sensing and pose-estimation latency separately.

5. Unified Absolute-Pose Benchmark

Forecasting absolute human poses can be important for a variety of applications such as motion planning, virtual reality or human-robot interaction. If a person moves around, their absolute position in the world can contain important information that is not captured by relative pose forecasts, for example future distances between human joints and objects or devices in the environment, and therefore would require an additional trajectory prediction model.

Rather than building a more complex two-model system that stacks trajectory and relative forecasting models, and then also their prediction errors, this work evaluates whether the already existing relative forecasting algorithms can be adapted to directly predict absolute poses as well.

To use the relative algorithms with absolute poses, the input pipeline was replaced. Instead of reusing the preprocessed data without global motion, which all approaches took from [24], the absolute poses were extracted directly from the *Human3.6M* dataset. This avoids potential sources of error introduced by repeated conversions between absolute and relative pose representations and additionally simplifies the overall data processing pipeline.

To improve the training generalization, the input and target pose sequences were moved to a central location by subtracting the mid-hip joint of the last input pose from all joint coordinates. For inference, the already known last input hip position is added again, to obtain global coordinates. The subtraction ensures that the model learns location-independent motion patterns, even though it might be interesting in some usecases to keep this information, for example, if movements only happen at certain locations.

For the main comparison, the *Human3.6M* [14] dataset is used, downsampled to 25Hz by skipping every second step, as in previous works. The models are trained on 50 input and 25 output timesteps, resulting in 1s of motion forecast, and only use 13 joints (2 hips, 2 shoulders, 1 nose, 2 knees, 2 ankles, 2 elbows, 2 wrists) to simplify usage with other datasets that have fewer ground-truth joint labels. The *MPJPE* is calculated action-agnostically by averaging over all dataset samples. All models are trained from scratch and evaluated on the same data, using released default hyperparameters whenever possible (required compatibility modifications are documented in Appendix C, and code and configuration files are included in the open-source repository).

Method	Type	Size	MPJPE	FPS	FCE	FADE
<i>Repeat last frame</i>	-	-	301	-	-	301
<i>Last delta average</i>	-	-	264	-	-	264
<i>Ridge Regression</i>	-	-	185	-	-	185
TBiFormer '2023	abs	5.64M	301	293	7	302
PGBIG '2022	rel	2.65M	279	35	57	287
GMFnet '2024	rel	11.3M	279	51	39	284
SPGSN '2022	rel	5.21M	262	19	105	276
STSGCN '2021	rel	202K	275	1405	1	275
HisReplItself '2020	rel	2.93M	233	142	14	235
DePOSit '2023	rel	1.17M	197	6	333	230
GCNNext '2024	rel	4.95M	175	38	53	180
siMLPe '2022	rel	129K	178	158	13	179
SoMoFormer '2022	abs	4.88M	179	586	3	179
MotionMixer '2022	rel	37.2K	168	632	3	168
IAFormer '2024	abs	2.53M	158	28	71	164
AuxFormer '2023	rel	1.27M	164	403	5	164
JRTransformer '2023	abs	3.74M	162	595	3	162
DeformMLP '2024	rel	1.29M	159	512	4	159
EMPMP '2025	abs	314K	150	215	9	151
EqMotion '2023	rel	3.36M	148	45	44	151

Table 2. Unified comparison of *absolute* and *relative* methods in the task of *absolute* pose forecasting on *Human3.6M*. *MPJPE* and *FADE* are measured at timestep 1000 *ms*. The error metrics are in millimeters. *FPS* is tested on a computer with an AMD-9900X and a single Nvidia-RTX4080. A batch-size of 1 was used for testing. Note that the size and speed of some models depends on the number of input and output timesteps (here: 50 in, 25 out).

Table 2 shows that several relative forecasting models can be adapted successfully to absolute pose forecasting. A new static approach was added for comparison as well, *last delta average* repeats the average of the last input pose delta, so all joints keep moving in the direction of the last general movement (the average delta is used to preserve the human shape over the time). As an additional classic method baseline, a simple *Ridge Regressor* was trained.

An interesting finding is that most models using graph neural networks (*HisReplItself*, *STSGCN*, *SPGSN*, *PGBIG*, *GMFnet*) perform worse in this unified setting than the others using a different network structure. This suggests that conclusions drawn from the original protocols may not directly transfer to this absolute-forecasting protocol, but it should not be read as evidence that graph-based models are generally inferior. At the same time, the results show that the general idea of switching the task of the relative models was successful, because some of those adapted implementations obtain lower errors than several evaluated absolute-pose implementations, including *SoMoFormer*, *JRTransformer*, and *EMPMP*.

6. Cross-Domain Sequence Baselines

The results in Table 2 show that diverse architectures obtain similar results and that several relative-pose models transfer to the absolute-pose protocol. From this observation, it remains unclear to what extent pose-forecasting performance depends on specialized architectures. This motivated testing

whether mature generic sequence architectures, originally introduced years earlier, can match or outperform recent pose-specific models after minimal adaptation. The goal was not to identify the best possible architecture, but to test whether current baseline comparisons are incomplete.

6.1. Scriboora

The idea behind *Scriboora* is the question: *Can speech-to-text models be retasked as well?* At a high level, both speech recognition and pose forecasting transform an observed numerical sequence into a target sequence, although the target spaces differ and are either character-tokens or joint coordinates. Furthermore, speech recognition provides many mature sequence models with diverse architectures that emphasize efficient inference while capturing local temporal patterns and long-range sequence structure, all of which are relevant to pose forecasting. For the experiments with different speech-to-text models (*DeepSpeech* [26], *QuartzNet* [18], *Conformer* [10], *Squeezeformer* [16]), the implementations from *Scribosermo* [2] were used, which also motivated the first part of the paper title, *Scriboora*.

6.2. Performance Results

The main change needed was to adapt the data pipeline, so that motion instead of audio sequences could be loaded. The models themselves only required minor changes. Appendix E provides details on the model changes and training procedure. Table 3 shows that the converted speech models are strong baselines under the unified protocol, with several matching or surpassing specialized models from Table 2. The *Conformer* models combine convolutional, linear, and attention layers to capture both local and global patterns and are therefore among the leading approaches in speech recognition. This relatively generic architecture also appears to adapt well to the new task.

Method	Size	MPJPE	FPS	FCE	FADE
DeepSpeech '2014	2.35M	198	653	3	198
QuartzNet '2020	3.17M	168	1300	2	168
Squeezeformer '2022	3.31M	149	960	2	149
Conformer '2020	3.22M	147	1049	2	147
MotionConformer	9.23M	143	929	2	143

Table 3. Results of converted speech models on *Human3.6M*.

As can be seen in Table 3 as well, most models are very fast, so this leaves room to further increase the size of future models. With that in mind, the *Conformer* model was further improved by increasing the layer dimensions, and as another change, because the motion sequence length is notably shorter than the average audio sequence length, the time downsampling at the start of the model was moved to the end of the model, so that information is retained longer in the model (see Appendix E). The resulting model is called

MotionConformer in the following. Although it is the second-largest model overall, the training time of 1h on the Nvidia-RTX4080 is at the lower end among the compared architectures. As shown in Table 3, it obtains the lowest reported error among the models evaluated under the unified protocol, while maintaining a very high inference speed as well.

7. Further Forecast Benchmarks

The previous benchmark evaluated a broad set of methods on the unified *Human3.6M* protocol. To test whether the main trends persist beyond this dataset, this section reports additional experiments on *CMU-MoCap* and *CHI3D*. Because retraining every audited model on every additional dataset would be computationally expensive, these experiments focus on the strongest methods from the unified benchmark and on representative baselines.

7.1. Single-person Forecasts

For the single-person evaluation, the *CMU-MoCap* [12] dataset was selected as an additional absolute-pose forecasting benchmark. The labels were extracted from the original dataset again, and are different to the ones used in *MRT*, *SoMoFormer*, *JRTransformer* and *EMPMP*. The processing script of *MRT* adds a randomly chosen global position offset, and also combines single actor sequences with other sequences to get sequences containing three actors, which are then cut to fixed sized 4s sub-sequences. Since for this experiment only single-actor sequences are evaluated, sequences with two actors are split into two. The performance of the models trained and tested on *CMU-MoCap* can be found in Table 4. The relative performance between the models is similar to the one on *Human3.6M*, with *MotionConformer* again obtaining the lowest errors.

Method	Size	MPJPE 400 / 1000	FPS	FADE 400 / 1000
<i>Repeat last frame</i>	-	201 / 408	-	201 / 408
<i>Last delta average</i>	-	167 / 398	-	167 / 398
<i>Ridge Regression</i>	-	130 / 296	-	130 / 296
JRTransformer	3.79M	99.0 / 245	610	99.4 / 245
DeformMLP	1.29M	98.3 / 242	462	98.8 / 243
EqMotion	3.37M	90.6 / 217	41	96.1 / 222
EMPMP	447K	93.1 / 216	213	94.2 / 217
MotionConformer	9.23M	88.6 / 201	909	88.8 / 201

Table 4. Comparison of single person forecasting on *CMU-MoCap* dataset. The models were trained to predict a maximum time-range of 1000ms, the forecast error was measured at the timesteps 400ms and 1000ms.

The second experiment in Table 5 evaluates the models on a longer time-range of 3000 ms. Especially for systems with longer reaction times, like robots or cars, longer forecasts can help to plan their actions better. Here a larger gap between the models can be observed. Specifically at the longer time-range, *DeformMLP* also seems to suffer from the too short

10 possible input timesteps, whereas the other models could predict the motion from 180 input frames. Apart from that, the relative performance between the models is similar to the previous experiments again.

Method	Size	MPJPE 1000/2000/3000	FPS	FADE 1000/2000/3000
<i>Repeat last frame</i>	-	356 / 562 / 706	-	356 / 562 / 706
<i>Last delta average</i>	-	373 / 773 / 1198	-	373 / 773 / 1198
<i>Ridge Regression</i>	-	276 / 507 / 692	-	276 / 507 / 692
DeformMLP	1.29M	241 / 454 / 656	226	242 / 455 / 657
JRTransformer	4.38M	255 / 446 / 602	562	255 / 446 / 602
EMPMP	3.91M	233 / 385 / 489	201	234 / 386 / 490
EqMotion	3.42M	218 / 354 / 469	13	235 / 368 / 481
MotionConformer	9.23M	211 / 351 / 462	777	211 / 351 / 462

Table 5. Comparison of single person forecasting on *CMU-MoCap* dataset. The models were trained to predict a maximum time-range of 3000ms, the forecast error was measured at 1000ms, 2000ms and 3000ms. Again, twice the output duration was used as input duration. The reason for the difference of *repeat last frame* method at timestep 1000ms to the table before is the windowing of the dataset. Here, the window is much larger, resulting in fewer samples, because the windows start later and end earlier.

7.2. Multi-person Forecasts

The previous benchmarks in this work focused on single-person forecasting. In many applications, like human-robot interactions, there is only one main actor, or the persons are unrelated to others. In such cases, the forecasts can be run independently per person. However, in some cases the persons can interact closely and influence each other’s movements. Therefore, the following small experiment is intended to show the support for multi-person inputs.

In the previous works of *JRTransformer* [42], and *EMPMP* [43] the multi-person dataset *CMU-MoCap* followed the preprocessing of *MRT* [36], including the scale problems, and partially missing preprocessing source-codes explained in more detail in Appendix B. Besides that, the multi-person interactions are partially synthetic, because the dataset combines different sequences to three person sequences, as mentioned earlier as well. Therefore, *CHI3D* [7] is used instead for the experiment. It shows two persons interacting closely with each other, and has a recording setup very similar to the one of the *Human3.6M* dataset, using four cameras in the room edges.

The results in the upper part of Table 6 show that the single-person models of *EqMotion* and *MotionConformer* can successfully be extended to multi-person forecasts. Technically the model inputs just need to be extended to 26 instead of the original 13 joints. The general idea behind this simple approach is, that if the model can learn some connection between arm and leg movements of one person from input coordinates, it should be able to learn some connection between two hands of different persons as well. In the paper of *IAFormer* [39] this dataset was already used as well,

and a *MPJPE* of 218 *mm*, and 233 *mm* for *TBiFormer* were reported. The results could not be replicated though, for unknown reasons there was no notable learning progress. However, the reported *MotionConformer* error is lower than their original published results.

Method	Size	MPJPE	FPS	FCE	FADE
<i>Repeat last frame</i>	-	272	-	-	272
<i>Last delta average</i>	-	318	-	-	318
<i>Ridge Regression</i>	-	277	-	-	277
EqMotion	3.37M	234	16	125	249
EMPMP	314K	227	209	10	228
MotionConformer	10.2M	207	853	2	207
MC (<i>chi3d+h36m</i>)	10.2M	191	853	2	191

Table 6. Forecasting close two-person interactions on *CHi3D*, errors at timestep 1000 *ms*.

Since the dataset is very small (the trainset only has around 0:13h of movements), especially the larger models suffer a bit from the data sparsity. Besides the iterative fine-tuning on live data, which will be explored in more detail in the next section, one also could enhance the training dataset with synthetically created pair motions, or by adding similar single person motions to it. In a short experiment, which simply added the *Human3.6M* dataset to the training sequences, the results of the *MotionConformer* in the lower part of Table 6 notably improved. This also shows transfer capabilities of the model from single to multi-person motions.

7.3. Further Suggestions

As a last note in this section, as a suggestion for future work, it should be beneficial to also add positions of objects and obstacles into the model inputs. The current forecasting datasets are normally object and obstacle free, but in realistic environments this is seldom the case. Often humans interact with objects as well, like in a soccer game in which the ball will notably guide the future movement direction, or try to avoid obstacles, like in buildings with doors. But this is out of scope for this work, and will be left for future research.

8. Noisy Pose Evaluation

While recent pose forecasting methods report strong results on clean motion-capture data, deployment systems often rely on poses estimated from images. The resulting detector noise can introduce spatially structured and temporally correlated errors that are not captured by clean-label evaluation.

8.1. Previous Approaches

With [6] and [28] there already are two works that investigated the impact of artificially generated noise. Both of them randomly dropped some of the joints and also added Gaussian noise to the joint positions. Only *DePOSit* [28] open-sourced their code for further usage. Dropping random joints is a type of noise that normally does not occur

in markerless settings, because most pose estimators always attempt to estimate joints that are not visible, without explicitly marking those estimated joints (which would allow dropping them afterward). Regarding the Gaussian noise, across multiple frames (as in the model inputs) random noise averages around the true location, while pose estimators rather have more structural errors (like forearm too short in all frames) in which the noise doesn't follow a Gaussian distribution. Due to those reasons, this work will investigate a more realistic type of error, namely the one caused by a preceding pose estimation network.

8.2. Detector-Noisy Data

To simplify the comparison between the performance with and without such noisy joint labels, a dataset is needed which also contains ground-truth labels. While the authors of [29] already have created a pose forecasting dataset with predicted joint labels (using *VoxelPose* [33]), it has no ground-truth labels. Furthermore, it contains almost no global movements of the persons as they are standing at a fixed position in front of a robot and mostly only move their arms.

Therefore, a new variant of the *Human3.6M* dataset is generated. The joints are predicted by a state-of-the-art multi-view multi-person model, namely *RapidPoseTriangulation* [3]. It achieves strong zero-shot performance on the previously unseen *Human3.6M* dataset of 52*mm* *MPJPE*. In the very rare cases (0.1%) that the person was not detected, or only with a score below a threshold (of 0.1 in a range from 0-1), the joints of this frame were replaced with the joints of a valid previous frame, mirroring a simple real-time tracking policy. In case multiple persons were detected, the one with the best score was kept. The pose prediction was very fast, with an average of 160 *FPS* on a single Nvidia-RTX4080 GPU. For consistency with the main benchmark, the data is subsampled to 25 frames per 1000*ms* again.

The reason for choosing a multi-view pose estimator is that for most practical applications, multi-view systems are expected to be a notably better choice than single-view systems. While they need some extra setup effort, this is comparably small, and avoiding/reducing errors in the input skeleton sequences, caused for example by occlusions, depth ambiguities, or image-to-pose estimation errors in one view, far outweighs the small additional calibration effort. Especially in professional or semi-professional setups in which a high pose and forecast accuracy is important. This was investigated in more depth in [4], which showed, for the related task of skeleton-based action recognition, that by switching from single-view to multi-view poses, the action classification error of multiple state-of-the-art methods could be significantly reduced, and also described why they are preferable to lifted 2D-to-3D skeletons from single-view inputs. Since the tasks are very similar, it seems reasonable to argue that those results transfer to skeleton-based motion forecasting as well.

8.3. Noise Adaptation Experiments

To approximate a deployment scenario in which a forecasting model is available before being adapted to a new application setup, the models are first pretrained on a mixture of external motion-capture datasets (here *CMU-MoCap* [12] and *BMLmovi* [8], *BMLrub* [32], *KIT* [23] from the *AMASS* [22] collection, which together contain around 11:30h of motions). The 13 joints from before are used again, but in the case of *CMU-MoCap* the *nose* is replaced with the *upper-head* joint position since no *nose* joint exists in this dataset. The original frame-rates of those datasets are 120fps or 30fps, but to match the target dataset, forecast is done with 25 steps. This changes the effective motion speed for some source datasets, but it is expected that the model should still be able to learn the general motion patterns.

Before evaluating detector noise, Table 7 reports clean-input zero-shot transfer to *Human3.6M*. Generalization to unseen datasets is a central requirement for practical forecasting models, and the results are close to the supervised *Human3.6M* benchmark in Table 2. This indicates that the pretrained models have strong cross-dataset generalization capability. Furthermore, the results provide a clean-input reference for measuring the degradation caused by detector-generated poses.

Method	MPJPE		
	40ms	400ms	1000ms
EMPMP	4.3	59.7	165
EqMotion	4.0	52.0	156
MotionConformer	4.0	51.6	149

Table 7. Zero-shot transfer capabilities from a large mixture of datasets to *Human3.6M*, at the given timesteps.

Table 8 evaluates the same pretrained models when the input poses are produced by the 3D pose-estimation network. The detector-target columns compare forecasts against future detector-generated poses, corresponding to the error measurable in a deployed system without motion-capture ground truth. The ground-truth-target columns compare the same forecasts against clean future poses, measuring the true error with respect to the actual human motion, including the up-

Method	detector target		ground-truth target	
	400ms	1000ms	400ms	1000ms
EqMotion	513	842	532	856
EMPMP	100	215	123	229
MotionConformer	94.2	214	118	228

Table 8. Influence of skeleton joints generated by a 3D-pose-estimation network on model performance, measured as MPJPE in millimeters at the given timesteps. The left section uses the network predictions as inputs and targets (which would be the measurable error in live systems), and the right one the predictions for input and ground-truth labels as targets (which would be the real error).

stream pose-estimator error (here on average 63 mm at time zero). Compared with the clean-input reference in Table 7, detector-generated inputs cause a substantial performance degradation, and the true error is consistently higher than the detector-target error observable from detector outputs alone.

Since detector outputs can be collected without additional motion-capture labels, the forecasting model can be adapted to the target application setup using detector-generated pose sequences as pseudo-labels. The upper block of Table 9 reports this adaptation for the pretrained *MotionConformer*. This setting simulates a deployed system that has collected observations from its target environment (here for around 1:55h) and finetunes the forecasting model with them. Following this approach, the detector-target error improves strongly (by 25.7%), and the ground-truth-target error also decreases (by 22.4%), indicating that pseudo-label adaptation reduces part of the detector-induced degradation. Although training and evaluating on clean motion-capture data still gives the best clean-input result (143 mm at 1000 ms), the adapted detector-trained model remains competitive under detector-target evaluation (159 mm at 1000 ms), suggesting that detector-generated pseudo-labels can be a viable training signal when clean labels are unavailable. Additional experiments with artificial Gaussian-noise pretraining are reported in Appendix F, showing that pretraining with artificial noise can improve robustness to detector-generated inputs but does not replace adaptation on the target detector outputs.

Method	detector target		ground-truth target	
	400ms	1000ms	400ms	1000ms
MotionConformer	66.9	159	94.8	177
EMPMP	76.4	179	104	195
EqMotion	68.5	168	96.1	185
MotionConformer	69.1	165	98.3	183

Table 9. Improvements through pseudo-label training. Finetuning the pretrained model on noisy predictions in the upper part, training all models from scratch in the lower part. Columns are separated the same as in Table 8

Because finetuning implementations were not available for the other two methods, the lower block of Table 9 reports the performance after training all models from scratch on the detector-generated dataset. In terms of MPJPE the results of *EqMotion* and *MotionConformer* are similar, but *MotionConformer* retains the runtime advantage observed in the previous section. The comparison between the two *MotionConformer* rows shows that pretraining on a large motion dataset is beneficial, although the final accuracy improvement over from-scratch training is moderate. Finetuning also converges faster, and a generic pretrained model is typically required before target-environment observations in a practical deployment scenario can be collected.

9. Limitations

The unified *Human3.6M* benchmark is intended as a controlled reference protocol for future absolute pose forecasting evaluations. Given the reproducibility and comparability issues found in existing benchmarks, it is proposed as a more reliable basis for future comparisons rather than merely as an additional leaderboard. At the same time, its scope is protocol-specific: the primary benchmark uses a 13-joint, single-person, absolute-coordinate representation together with fixed preprocessing, windowing, and metric choices. Results should therefore not be interpreted as numerically interchangeable with protocols that use different joint sets, coordinate systems, or evaluation metrics. Additional experiments on *CMU-MoCap* and *CHI3D*, as reported in Section 7, can provide complementary cross-dataset evaluations and should be used alongside the primary *Human3.6M* protocol to assess whether observed performance trends generalize beyond a single dataset.

The benchmarks generally follow the common practice of the evaluated literature and report one training run per method. This makes a broad audit of many third-party implementations feasible, but it does not measure training variability of all methods. Such scores are thus best interpreted as reproducible artifact-level comparisons under the unified protocol. For the best three methods, ablation experiments with multiple training runs with different random seeds were therefore performed and reported in Appendix E.2, and show a low variance of the reported scores. Even across the other datasets in Section 7 (which at the same time use different seeds) the ranking stays the same, which provides further support for the overall ranking stability, although it does not establish it conclusively.

The cross-domain baseline study is also not an exhaustive architecture search, but rather tests whether mature generic sequence models are missing from current baseline comparisons. For the proposed *MotionConformer* models, trained checkpoints are released together with the evaluation code, making the reported artifact performance directly reproducible. For third-party methods, reproducibility depends on the artifacts released by the original authors and on the compatibility of their training code with the unified protocol. Future benchmark reports should therefore additionally include seed variation results when the claim concerns method-level superiority rather than the performance of a released checkpoint, especially when numerical margins are comparatively small.

The detector-noisy *Human3.6M* variant measures robustness to one multi-view pose-estimation pipeline. The selected detector was reported in its original paper to perform best across multiple datasets that were unseen during its training, making it a strong representative source of detector-induced noise. Nevertheless, other pose estimators may induce different spatial and temporal error distributions. Ad-

ditionally, most commonly used datasets are captured in controlled laboratory settings, that might not perfectly reflect the conditions of real-world deployment. Therefore future work should evaluate whether the observed degradation and recovery behavior generalizes across detector families, and whether it persists under less controlled capture conditions. However, such experiments are outside the scope of this work. The proposed protocols nevertheless establish a reproducible and comparable framework for systematically evaluating the effects of detector noise under controlled conditions, while providing a common basis for future extensions.

10. Conclusion

This work re-examined human pose forecasting with a focus on evaluation reliability, model assumptions, and deployment realism. The reproducibility audit uncovered several issues in prior evaluation code, preprocessing pipelines, artifact availability, and metric definitions. By re-evaluating recent approaches under a unified absolute-pose protocol, this work provides a more controlled reference point for future comparisons.

The results also question whether progress in pose forecasting should be attributed primarily to specialized forecasting architectures. By viewing pose forecasting as a sequence-to-sequence problem, architectures originally developed for speech recognition were adapted to this setting. The adapted *Conformer* models, and especially the optimized *MotionConformer*, obtained the lowest reported errors under the unified protocol while maintaining high inference throughput. This suggests that mature generic sequence models should be treated as serious baselines, and that architectural claims in pose forecasting should be evaluated under stricter and more comparable protocols.

To move evaluation closer to deployment conditions, this work introduced latency-aware metrics and a paired clean/detector-noisy *Human3.6M* variant. The noisy-input experiments show that models trained and evaluated only on clean motion-capture data can suffer a substantial performance drop when their inputs come from a pose estimator. At the same time, adapting the forecasting model on detector-generated pose sequences can recover part of this degradation without requiring additional ground-truth annotations.

Overall, this work shows that rethinking human pose forecasting requires more than proposing new architectures. Reliable progress depends on reproducible artifacts, clearly specified preprocessing, consistent metrics, strong generic baselines, and evaluation under realistic input conditions. To support future work in this direction, all code, preprocessing scripts, trained *MotionConformer* checkpoints, and evaluation tools are made publicly available.

References

- [1] Vida Adeli, Mahsa Ehsanpour, Ian Reid, Juan Carlos Niebles, Silvio Savarese, Ehsan Adeli, and Hamid Rezatofighi. Tripod: Human trajectory and pose dynamics forecasting in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13390–13400, 2021. 2, 11, 12
- [2] Daniel Bermuth, Alexander Poeppel, and Wolfgang Reif. Scribormo: Fast Speech-to-Text models for German and other Languages. *arXiv preprint arXiv:2110.07982*, 2021. 4
- [3] Daniel Bermuth, Alexander Poeppel, and Wolfgang Reif. RapidPoseTriangulation: Multi-view Multi-person Whole-body Human Pose Triangulation in a Millisecond. *arXiv preprint arXiv:2503.21692*, 2025. 6, 15
- [4] Daniel Bermuth, Alexander Poeppel, and Wolfgang Reif. Skarimva: Skeleton-based action recognition is a multi-view application. In *2026 IEEE 20th International Conference on Automatic Face and Gesture Recognition (FG)*, pages 1–5. IEEE, 2026. 6
- [5] Arij Bouazizi, Adrian Holzbock, Ulrich Kressel, Klaus Dietmayer, and Vasileios Belagiannis. MotionMixer: MLP-based 3D Human Body Pose Forecasting. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 791–798. International Joint Conferences on Artificial Intelligence Organization, 2022. 2, 11, 12
- [6] Qiongjie Cui and Huaijiang Sun. Towards accurate 3d human motion prediction from incomplete observations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4801–4810, 2021. 2, 6
- [7] Mihai Fieraru, Mihai Zanfir, Elisabeta Oneata, Alin-Ionut Popa, Vlad Olaru, and Cristian Sminchisescu. Three-dimensional reconstruction of human interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7214–7223, 2020. 5
- [8] Saeed Ghorbani, Kimia Mahdavian, Anne Thaler, Konrad Kording, Douglas James Cook, Gunnar Blohm, and Nikolaus F Troje. MoVi: A large multi-purpose human motion and video dataset. *Plos one*, 16(6):e0253157, 2021. 7
- [9] Francesco Giuliani, Irtiza Hasan, Marco Cristani, and Fabio Galasso. Transformer networks for trajectory forecasting. In *2020 25th international conference on pattern recognition (ICPR)*, pages 10335–10342. IEEE, 2021. 11
- [10] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. *arXiv preprint arXiv:2005.08100*, 2020. 4
- [11] Wen Guo, Yuming Du, Xi Shen, Vincent Lepetit, Alameda-Pineda Xavier, and Moreno-Noguer Francesc. Back to MLP: A Simple Baseline for Human Motion Prediction. *arXiv preprint arXiv:2207.01567*, 2022. 2, 11, 12
- [12] Carnegie Mellon University: <http://mocap.cs.cmu.edu/>. CMU Graphics Lab Motion Capture Database. 2, 5, 7, 11
- [13] Haitao Huang, Chi-Man Pun, Haolun Li, Mengqi Liu, Jian Xiong, and Hao Gao. DeformMLP: dynamic large-scale receptive field MLP networks for human motion prediction. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5200–5204. IEEE, 2024. 2, 11, 12
- [14] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2014. 2, 3, 11
- [15] Jaewoo Jeong, Daehee Park, and Kuk-Jin Yoon. Multi-agent long-term 3d human pose forecasting via interaction-aware trajectory conditioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1617–1628, 2024. 2, 12, 13
- [16] Sehoon Kim, Amir Gholami, Albert Shaw, Nicholas Lee, Karttikeya Mangalam, Jitendra Malik, Michael W Mahoney, and Kurt Keutzer. Squeezeformer: An efficient transformer for automatic speech recognition. *Advances in Neural Information Processing Systems*, 35:9361–9373, 2022. 4
- [17] Parth Kothari, Sven Kreiss, and Alexandre Alahi. Human trajectory forecasting in crowds: A deep learning perspective. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):7386–7400, 2021. 11
- [18] Samuel Kriman, Stanislav Beliaev, Boris Ginsburg, Jocelyn Huang, Oleksii Kuchaiev, Vitaly Lavrukhin, Ryan Leary, Jason Li, and Yang Zhang. Quartznet: Deep automatic speech recognition with 1d time-channel separable convolutions. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6124–6128. IEEE, 2020. 4
- [19] Jinkai Li, Jinghua Wang, Xin Wang, Liang Yan, and Yong Xu. Component-wise self-correction network for human motion prediction. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2025. 2, 12
- [20] Maosen Li, Siheng Chen, Zijing Zhang, Lingxi Xie, Qi Tian, and Ya Zhang. Skeleton-parted graph scattering networks for 3d human motion prediction. In *European Conference on Computer Vision*, pages 18–36. Springer, 2022. 2, 11, 12
- [21] Tiezheng Ma, Yongwei Nie, Chengjiang Long, Qing Zhang, and Guiqing Li. Progressively generating better initial guesses towards next stages for high-quality human motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6437–6446, 2022. 2, 11, 12
- [22] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of Motion Capture as Surface Shapes. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 7
- [23] C. Mandery, Ö. Terlemez, M. Do, N. Vahrenkamp, and T. Asfour. The KIT whole-body human motion database. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 329–336, 2015. 7
- [24] Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2891–2900, 2017. 2, 3, 11, 12, 13
- [25] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose esti-

- mation. In *Proceedings of the IEEE international conference on computer vision*, pages 2640–2649, 2017. [2](#), [11](#)
- [26] Mozilla. Project DeepSpeech, 2021. [accessed 26-February-2021]. [4](#)
- [27] Xiaogang Peng, Siyuan Mao, and Zizhao Wu. Trajectory-aware body interaction transformer for multi-person pose forecasting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 17121–17130, 2023. [2](#), [12](#), [13](#)
- [28] Saeed Saadatnejad, Ali Rasekh, Mohammadreza Mofayez, Yasamin Medghalchi, Sara Rajabzadeh, Taylor Mordan, and Alexandre Alahi. A generic diffusion-based approach for 3D human pose prediction in the wild. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8246–8253. IEEE, 2023. [2](#), [6](#), [11](#), [12](#)
- [29] Alessio Sampieri, Guido Maria D’Amely di Melendugno, Andrea Avogaro, Federico Cunico, Francesco Setti, Geri Skenderi, Marco Cristani, and Fabio Galasso. Pose forecasting in industrial human-robot collaboration. In *European Conference on Computer Vision*, pages 51–69. Springer, 2022. [6](#)
- [30] Junyu Shi, Jianqi Zhong, Zhiqian He, and Wenming Cao. Gradient multi-foci networks for 3D skeleton-based human motion prediction. *Neural Computing and Applications*, 36 (24):14627–14642, 2024. [2](#), [11](#), [12](#)
- [31] Theodoros Sofianos, Alessio Sampieri, Luca Franco, and Fabio Galasso. Space-Time-Separable Graph Convolutional Network for Pose Forecasting, 2021. [2](#), [11](#), [12](#)
- [32] Nikolaus F. Troje. Decomposing Biological Motion: A Framework for Analysis and Synthesis of Human Gait Patterns. *Journal of Vision*, 2(5):2–2, 2002. [7](#)
- [33] Hanyue Tu, Chunyu Wang, and Wenjun Zeng. Voxelpose: Towards multi-camera 3d human pose estimation in wild environment. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 197–212. Springer, 2020. [6](#)
- [34] Edward Vendrow, Satyajit Kumar, Ehsan Adeli, and Hamid Rezatofighi. SoMoFormer: Multi-Person Pose Forecasting with Transformers. *arXiv preprint arXiv:2208.14023*, 2022. [2](#), [11](#), [13](#)
- [35] Timo Von Marcard, Roberto Henschel, Michael J Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *Proceedings of the European conference on computer vision (ECCV)*, pages 601–617, 2018. [2](#), [11](#)
- [36] Jiashun Wang, Huazhe Xu, Medhini Narasimhan, and Xiaolong Wang. Multi-person 3D motion prediction with multi-range transformers. *Advances in Neural Information Processing Systems*, 34:6036–6049, 2021. [2](#), [5](#), [12](#), [13](#)
- [37] Xinshun Wang, Qiongjie Cui, Chen Chen, and Mengyuan Liu. Gnext: Towards the unity of graph convolutions for human motion prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 5642–5650, 2024. [2](#), [11](#), [12](#)
- [38] Mao Wei, Liu Miaomiao, and Salzmann Mathieu. History Repeats Itself: Human Motion Prediction via Motion Attention. In *ECCV*, 2020. [2](#), [11](#), [12](#)
- [39] Peng Xiao, Yi Xie, Xuemiao Xu, Weihong Chen, and Huaidong Zhang. Multi-person Pose Forecasting with Individual Interaction Perceptron and Prior Learning. In *European Conference on Computer Vision*, pages 402–419. Springer, 2024. [2](#), [5](#), [12](#)
- [40] Chenxin Xu, Robby T Tan, Yuhong Tan, Siheng Chen, Xinchao Wang, and Yanfeng Wang. Auxiliary tasks benefit 3d skeleton-based human motion prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9509–9520, 2023. [2](#), [11](#), [12](#)
- [41] Chenxin Xu, Robby T Tan, Yuhong Tan, Siheng Chen, Yu Guang Wang, Xinchao Wang, and Yanfeng Wang. EqMotion: Equivariant Multi-agent Motion Prediction with Invariant Interaction Reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1410–1420, 2023. [2](#), [11](#), [12](#)
- [42] Qingyao Xu, Weibo Mao, Jingze Gong, Chenxin Xu, Siheng Chen, Weidi Xie, Ya Zhang, and Yanfeng Wang. Joint-relation transformer for multi-person motion prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9816–9826, 2023. [2](#), [5](#), [12](#), [13](#)
- [43] Yuanhong Zheng, Ruixuan Yu, and Jian Sun. Efficient Multi-Person Motion Prediction by Lightweight Spatial and Temporal Interactions. *arXiv preprint arXiv:2507.09446*, 2025. [2](#), [5](#), [12](#), [13](#)

A. Related Work

This section provides a more detailed overview of the aforementioned related work methods evaluated in the experiments of this paper. The selection covers the main approaches identified in the literature, with the majority having been published at top-tier conferences. The key architectural aspects of each method are described below.

A.1. Relative Forecasting Approaches

Most relative approaches build upon the source-codes from [24, 25] and reuse their preprocessed datasets and evaluation metrics. The default dataset is *Human3.6M* [14]. In the commonly used subset, 7 actors are performing 15 different actions. The actors are represented as skeletons with 32 joints in total, which in the case of the preprocessed dataset, are represented as exponential maps, from which the 3D coordinates are computed. From the seven actors, the subject *S11* is used for validation, and *S5* is used for testing. The dataset was captured at a frame rate of $50Hz$, but usually every second frame is skipped, and only 25 future frames are predicted for the next second.

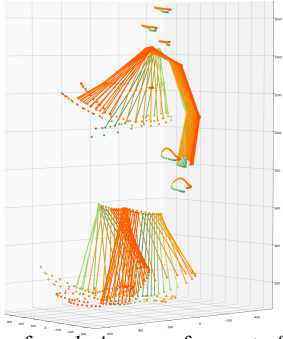


Figure 2. Example of a *relative* pose forecast of a walking person. The green skeletons visualize the input sequence of the prediction model, and the red ones the predicted future poses. Only the predicted joints are visualized, therefore this person does not have a hip (it is fixed to the same spot).

Out of the 32 joints 22 joints are selected for training (see Figure 2), the others either contain the static hip joints or are duplicated joints. To calculate the resulting *mean per joint position error (MPJPE)* according to the default protocol, the 32 target joints are copied and the values of the 22 training joints are replaced with the model predictions. Then the mean error over all 32 joints is calculated. For each action 256 randomly sampled sequences are used.

The relative approaches investigated in this paper mostly use different network architectures. *HisReplItself* [38] uses a motion-attention concept to attend to historical motion sub-sequences, which are then used by a graph convolution network to predict future poses. *STSGCN* [31] only uses a graph convolution network, which can separate between the space and time axis of the motions. *SPGSN* [20]

splits the skeleton into separate parts and analyzes both, parts and the complete skeleton model to extract features and predict future poses. *PGBIG* [21] uses a multi-stage graph network in which initial guesses are progressively improved until the final output. *MotionMixer* [5] relies solely on multi-layer perceptrons, which alternate between using spatial and temporal channels as inputs. *siMLPe* [11] uses a similar, but much simpler architecture, which in its core has only alternating fully-connected and layer-norm layers. *EqMotion* [41] distinguishes between geometric and pattern features and additionally creates an interaction graph for the inputs. The model is based on fully-connected layers and has multiple connections between the geometric and pattern layers. *DePOSit* [28] uses a diffusion model to simultaneously predict future poses and de-noise them. In some of their experiments, the prediction was done in two steps with two models, one for the short-term future as an intermediate step, and the second that uses the predictions from the first as input for the complete prediction. Its architecture was directly designed to handle noisy inputs as well. *AuxFormer* [40] uses a transformer-based architecture, which includes auxiliary tasks in its training, to improve the forecasting performance. Instead of just forecasting future poses, it also has network branches designed to repair artificially added noise and dropout errors. *GMFnet* [30] developed a two-stage model to improve the focus of attention layers for better feature extraction. *GCNext* [37] improves the design of graph convolution layers to allow them to calculate features across space, time and channels (joints) simultaneously. *DeformMLP* [13] uses a multi-layer perceptron architecture that splits inputs into non-overlapping blocks which are also mixed across space, time and channel dimensions.

A.2. Absolute Forecasting Approaches

Later on, there have been attempts to calculate absolute pose motions instead of relative poses. For applications requiring the full human movement, this has the benefit that the model outputs can be used directly, and do not need to be combined with models that focus on forecasting only the general motion trajectory, like [9, 17]. An earlier approach for this is *TRiPOD* [1], which also created a new benchmark. For 3D joint predictions they use the *3D Poses in the wild* (3DPW) dataset [35], which consists of 60 action sequences in a real world setting. Another dataset often used for absolute pose forecasting is *CMU-MoCap* [12]. The actors have been captured by twelve cameras in a $3m \times 8m$ room with a frame-rate of $120Hz$. Usually this rate is reduced to 15 frames per second by skipping over frames. In most scenes there is only one actor, but some contain a second actor as well.

SoMoFormer [34] was one of the earlier leading algorithms on both datasets and is also available open-source. The algorithm uses a transformer-based method, which uses a learned joint and person embedding to allow move-

ment predictions for multiple persons at the same time. *TBiFormer* [27] splits the persons into sequences of body parts and then learns self- and inter-person relations using those part features. *JRTransformer* [42] distinguishes between joint and relation features and has specialized layers for both, which are regularly fused with each other. *IAFormer* [39] focuses on the interaction between persons and tries to learn person-independent interaction features to especially improve multi-person forecasts. *T2P* [15] follows a coarse-to-fine approach, by first forecasting global trajectories, followed by conditioned local pose forecasts. *EMPMP* [43] proposes a lightweight dual branch architecture to learn local and global motions together with a cross-level interaction module to share information between the two branches.

B. Detailed Reproduction Notes

Before comparing all models on the same dataset, the first question was: *Can the results of the approaches described in their respective papers be reproduced?* What should have been a straightforward reproduction exercise instead revealed substantial issues in many of the audited methods. This appendix provides the detailed evidence behind the audit summary in Table 1. For each method, the reported result, available artifacts, reproduction outcome, and identified discrepancy are described.

B.1. Relative Forecasting Approaches

For relative forecasting methods, several approaches could be reproduced exactly or within small numerical differences, while others showed larger discrepancies.

STSGCN [31] had an error in their MPJPE calculation, which the authors mentioned on their linked repository. The corrected results could be reproduced here. *MotionMixer* [5] calculated both metrics, but also had an error in the old metric (at counting items for the average calculation), which could be corrected because they open-sourced the code, but it resulted in significantly lower performance. The reason why *SPGSN* [20] and *EqMotion* [41] had a lower performance is unclear. Similar reproduction difficulties were reported in public repository issues, but no resolved explanation was available at the time of the audit. With *DePOSit* [28] and *GMFnet* [30] the reason is unclear as well. The results of *EqMotion*, *AuxFormer*, *GMFnet* (400 ms) and *DePOSit* could be reproduced with their published checkpoints though. The repository of *IAFormer* [39] did not contain the code for the *Human3.6M* dataset, but their results on *CMU-MoCap* could be replicated. *CSCNet* [19] has no published source code. A comparison of all relative approaches, including the corrected results, can be found in Table 10. *Res. Sup.* [24] served as the basis on which the other works build upon.

Method	MPJPE (400/1000ms)	Replicated
<i>Repeat last frame</i>	88.2 / 136.6	-
<i>Ridge Regression</i>	76.5 / 128.2	-
<i>Res. Sup.</i> [24, 38] '2017	88.3 / 136.6	-
<i>IAFormer</i> [39] '2024	50.8 / 130.4	-
<i>MotionMixer</i> [5] † '2022	59.3 / 111.0	65.4 / 117.9
<i>STSGCN</i> [31] † '2021	38.3 / 75.6	67.5 / 117.0
<i>GMFnet</i> [30] '2024	- / 102.3	64.1 / 114.1
<i>SPGSN</i> [20] '2022	58.3 / 109.6	62.0 / 112.3
<i>HisReplself</i> [38] '2020	58.3 / 112.1	58.3 / 112.0
<i>AuxFormer</i> [40] '2023	- / 107.0	61.7 / 111.6
<i>PGBIG</i> [21] '2022	58.5 / 110.3	58.8 / 110.4
<i>EqMotion</i> [41] '2023	55.0 / 106.9	57.8 / 109.8
<i>GCNext</i> [37] '2024	56.4 / 108.7	57.5 / 109.8
<i>CSCNet</i> [19] '2025	57.4 / 109.4	-
<i>siMLPe</i> [11] '2022	57.3 / 109.4	57.6 / 109.6
<i>DePOSit</i> [28] '2023	- / 103.3	59.0 / 106.2
<i>DeformMLP</i> [13] '2024	57.9 / 105.5	57.5 / 105.3

Table 10. Comparing recent approaches for *relative* pose forecasting on *Human3.6M* dataset, averaged over all actions. The *mean per joint position error* (MPJPE) is measured in millimeters at two frames 400 ms and 1000 ms in the future (using the same model for both timesteps). Approaches marked with a † had errors in their evaluation code which were fixed in the *replicated* column.

B.2. Absolute Forecasting Approaches

For absolute forecasting methods, the main reproducibility problems were metric definitions, scale conversions, and timestamp conventions.

In the absolute pose forecasting benchmark of *TRi-POD* [1] a new metric called *Visibility-Ignored Metric* (VIM) was developed, which in the case of no invisible joints, should behave like MPJPE for each timestep ("This metric is the simple MPJPE metric except that the invisible joints (if exist) are not penalized and are simply discarded by considering truth"), but it has an error in its formula (see Equations (4) and (1)). Instead of calculating the distance in 3D-space and averaging over the joints, the distance is calculated in 3*N-dimensional joint space. It is measured in centimeters. To calculate VIM the train/test splits of 3DPW also need to be switched.

$$VIM_t = \sqrt{\sum_{i=1}^{joints} \left(\sum_{k=1}^3 (gt_{t,i,k} - pred_{t,i,k})^2 \right)} \quad (4)$$

For comparison on the *CMU-MoCap* dataset, it was stated in *SoMoFormer* that the same training recipe and evaluation metric was used as in *MRT* [36]. In the *MRT*-paper it was described that the error unit is 0.1 meters, which already includes a dataset scaling factor of 0.6, but following the source code of *MRT*, a different unit has to be used. The error has to be multiplied by $((10 * 3 / 1.8 * si2m) / 1.8) \approx 0.5226$ to get a unit of 1 meter, with $si2m = ((1.0 / 0.45) * 2.54 / 100.0) \approx 0.0564$, which following the dataset description, converts the scaled inches unit to meters. Following the large difference between the

results reported in the *MRT*-paper, and the ones for the *MRT*-model in the *SoMoFormer*-paper, the latter likely used a different metric or scale, but this could not be validated, because the source code was not completely open-sourced. The same also applies to *TBiFormer* [27], *JRTransformer* [42], and *EMPMP* [43]. Besides that, all papers reported different results for the same benchmark in their own replications, sometimes resulting in older models being better than the newer ones, or with large numeric error differences, which shows general replicability problems with those models (Table 11). One reason could be that often only parts of the code are published, so some steps might have been reimplemented differently. And for *T2P* [15] the environment could not be set up completely, therefore it was not evaluated in the main sections.

Method	Reporting Paper	MPJPE (1000ms / 3000ms)
HisRepItself	MRT [36]	1.05 / 1.58
	TBiFormer [27]	134 / 349
	SoMoFormer [34]	0.50 / 1.42
	JRTransformer [42]	14.9 / 30.7
MRT	MRT [36]	0.79 / 1.22
	TBiFormer [27]	148 / 352
	SoMoFormer [34]	0.46 / 1.11
	JRTransformer [42]	9.6 / 21.8
TBiFormer	EMPMP [43]	89 / 189
	TBiFormer [27]	118 / 329
SoMoFormer	EMPMP [43]	98 / 218
	SoMoFormer [34]	0.42 / 1.06
	JRTransformer [42]	10.7 / 22.5
JRTransformer	EMPMP [43]	84 / 202
	JRTransformer [42]	8.3 / 18.5
JRTransformer	EMPMP [43]	88 / 206

Table 11. Comparing reported results of recent approaches for *absolute* pose forecasting on the *CMU-MoCap* dataset proposed by *MRT* [36]. Even though the descriptions of the experiments are similar, the reported results differ substantially between papers, showing general replicability problems.

The authors of *TBiFormer* stated to use the preprocessing from *MRT* as well, but if that is correct, at least one of their results uses wrong timestamps. In the published source-code they used 50 input and 25 output frames for their 2s/1s experiment on *CMU-MoCap*. The base dataset has an original frame-rate of 120Hz, which is reduced to 30Hz by *MRT*. If they did not interpolate the frames (the preprocessing code was not published), the predicted time range is too short. The same problem could be found in the source-code of *IAFormer* and *T2P*, and the paper of *GMFnet*.

C. Converting relative to absolute Forecasters

To evaluate the released relative-pose forecasting implementations under the unified absolute-pose protocol, their input pipelines were replaced. Instead of reusing the preprocessed data without global motion, which all approaches

took from [24], the absolute poses were extracted directly from the *Human3.6M* dataset.

The goal was to make the methods comparable at the level of data, coordinates, joint definitions, and metrics while avoiding full reimplementations of each architecture. Whenever possible, models were trained with the same temporal setup: 50 input frames and 25 output frames for *Human3.6M*. Default hyperparameters from the released implementations were otherwise retained.

Some methods required compatibility modifications. For *DePOSit* the single-stage architecture without intermediate timesteps is used, similar to its original experiments with larger datasets, to significantly speed up training and inference (about $2\times$). *GMFnet* and *DeformMLP* had the 22 joints of the old dataset deeply integrated into their architecture, so to match their joint number, the 13 main joints are padded to 22. For the *MPJPE* calculation those padded joints were removed again from the prediction. For *DeformMLP*, the original 10-frame input length was retained because the 50-frame variant substantially increased model size and diverged in all attempted runs.

These modifications should be interpreted as minimal compatibility changes needed to evaluate available implementations under a shared protocol, not as an architecture-only comparison in which all methods were independently re-engineered to identical input assumptions.

D. FADE Reasoning

The *forecast after delay error (FADE)* is intended as a latency-aware approximation for fixed-horizon forecasting models. If a model requires an inference time of Δt , then a forecast requested for horizon t is only available after part of that horizon has already elapsed. In a deployment system, the relevant prediction is therefore closer to horizon $t + \Delta t$ than to horizon t . An exact evaluation of this delayed horizon would require each model to output predictions at delay-specific horizons, or would require retraining the models with longer output sequences. *FADE* avoids this additional retraining by estimating the error at the delayed horizon from the error measured at the original horizon.

This approximation assumes that the *MPJPE* grows approximately linearly in a small neighborhood after the evaluated timestep. Importantly, this is a local approximation around each forecast horizon, not a claim that the full *MPJPE* curve is globally linear over the complete forecast range. For a horizon t and delay Δt , *FADE* extrapolates the delayed error by extending the linear trend from zero error at time 0 to the measured error $MPJPE_t$ at time t .

Figure 3 visualizes this assumption for *MotionConformer*. The measured *MPJPE* values are shown together with short extrapolation segments. Each segment starts at a measured timestep and extends by one forecast step using the local linear increase that *FADE* would assume at that point. The

vertical difference between the end of the segment and the next measured *MPJPE* value therefore measures the local discrepancy between the *FADE* approximation and the actual forecast error curve.

For *Human3.6M* with 25 steps per second, this local discrepancy has a mean absolute error of 0.70 mm , and a maximum absolute error of 1.07 mm . For *CMU-MoCap* with 30 steps per second, the mean absolute error is 0.16 mm , and the maximum absolute error is 0.65 mm . These errors are small compared with the *MPJPE* values reported in the main experiments, supporting the use of *FADE* as a latency-aware approximation for the evaluated settings.

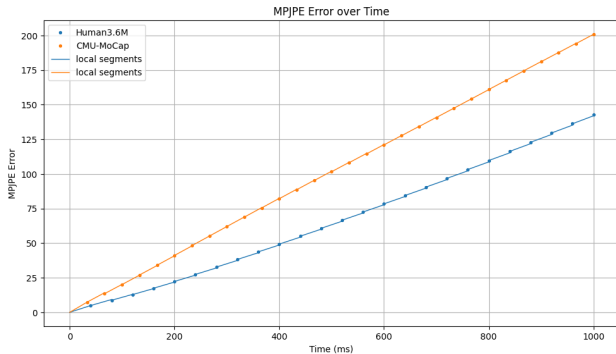


Figure 3. Measured *MPJPE* over time for *MotionConformer*, together with local one-step extrapolation segments used to validate the *FADE* approximation. Each segment starts at a measured timestep and follows the linear increase assumed by *FADE*. The small vertical offset between the segment endpoint and the next measured *MPJPE* value indicates the local approximation error.

E. Scriboora Details

This appendix provides implementation details for the cross-domain sequence baselines used in Section 6. The purpose of these models is not to claim that speech recognition is uniquely suited to pose forecasting, but to test whether mature sequence-to-sequence architectures from a neighboring domain already form strong baselines under the unified protocol. Speech-to-text models are useful for this stress test because they include diverse recurrent, convolutional, and attention-based architectures, while also being designed for efficient sequence processing.

E.1. Network Changes

Only minor model changes were required to adapt the speech-to-text architectures to pose forecasting. The data pipeline was replaced so that motion sequences rather than audio features were loaded, and the task-specific input and output layers were adjusted to predict joint coordinates. The hidden dimensions and filter shapes were modified to keep layer and model sizes comparable across variants. All *BatchNorm* layers were removed because they consistently degraded

training. For *Conformer* and *Squeezeformer*, the initial sub-sampling layers were changed from $4\times$ to $2\times$ temporal downsampling to match the required output sequence length. As with the other models in the unified benchmark, all *Human3.6M* models were trained from scratch.

E.2. Ablation Studies

As can be seen in Table 3, most converted speech models are very fast, which leaves room to increase the model size further. With that in mind, the *Conformer* model was improved by increasing the layer dimensions. As another change, because pose sequences are notably shorter than typical audio sequences, the time reduction at the start of the model was moved to the end, so that temporal information is retained longer in the network. For the same reason, larger speech models with stronger temporal compression were not considered here. The resulting model is called *MotionConformer*. Although it is the second-largest model overall, its training time of only $1h$ on the Nvidia-RTX4080 is on the lower end among the compared architectures. As shown in Table 12, it obtains the lowest reported error among the tested models while maintaining high inference speed.

Method	Size	MPJPE	FPS	FADE
Conformer	3.22M	147	1049	147
increased model size	8.41M	144	932	144
moved time reduction	3.59M	146	965	146
MotionConformer	9.23M	143	929	143
no spec augmentation		144		
no convolutional modules	7.89M	145	1024	145
xxl model size	122M	145	264	146
5-run averages				
EMPMP		152.3 ± 0.94		153.0
EqMotion		148.1 ± 1.19		151.4
Conformer (default)		147.6 ± 0.24		147.7
Conformer (increased size)		145.4 ± 0.47		145.6
MotionConformer		144.0 ± 0.45		144.2

Table 12. Implemented improvements and ablations of *MotionConformer*, tested on *Human3.6M*.

The ablations in Table 12 are primarily intended as exploratory checks, since some observed differences are small. The transition from the adapted *Conformer* to *MotionConformer* gives the clearest improvement, while the individual effects of the later ablations are less conclusive. *SpecAug*-style masking is a plausible augmentation for pose forecasting and should be investigated more closely in future work. It randomly masks contiguous parts of the input sequence along the temporal or channel axis, encouraging the model to handle missing timesteps or missing joints. The role of the *Conformer*-specific convolutional modules was also evaluated. Removing the convolutional modules from the *Conformer* blocks only slightly changes the result in this measurement. While this leaves open whether they provide a small benefit, it indicates that the convolutional components

are not the dominant source of the model’s performance. Further increasing the model size did not improve the performance, but only reduces the inference speed, which suggests that the model is already sufficiently large for this task. A five-seed rerun of the final *MotionConformer* configuration shows stable training. The gap between the mean and best checkpoint is small relative to the gap to the next-best evaluated models, and the best checkpoint is released for direct artifact-level reproducibility. The multi-seed-runs of the other models showed similar stability, and further strengthen the comparatively better performance of *MotionConformer*.

F. Artificial Noise Pretraining

As a lightweight robustness baseline, this appendix evaluates whether pretraining with synthetic coordinate noise improves performance under detector-generated inputs. Even though adding artificial Gaussian noise to the joint coordinates does not match the errors from a pose estimation system, it still can be used to prepare the forecasting network for such type of input. Finetuning the networks with random noise (*std*: 25mm, clip at 125mm) notably improves the zero-shot performance for longer (1000ms) predictions, as can be seen in Table 13. The training of *EqMotion* always diverged mid-training, so no results are available for this model here. As a first side note, further experiments with the other two models showed that in both cases, the performance on the clean ground-truth dataset slightly dropped after adding noise, but this is expectable because of the then lower train-test similarity. As a second side note, adding random position noise directly to the joint positions is preferable to using limb-rotation-based noise, because while axis noise mathematically also just changes the joint position in the end, it keeps the limb lengths correct, which is not true for detector noise, and therefore not desired.

Method	detector target		ground-truth target	
	400ms	1000ms	400ms	1000ms
EMPMP	82.5	188	108	204
MotionConformer	82.4	185	106	199

Table 13. Improvements through artificial noise pre-training. The left/right columns are separated into prediction/groundtruth targets, as in Table 8, showing the measurable/real error.

G. Impact of Joint Errors on Forecasting

Regarding the influence of the noise, the influence of missing joints should be neglectable (less than 0.2% of the joints have an error above 500mm, which is usually counted as “missing”). As only around 12% of the input joints have a displacement error above 10cm, the overall impact of very poor joints should not be the main extra MPJPE reason (as the experiments in the *RapidPoseTriangulation* [3] showed this level is also expectable (or even better) for other datasets). The additional error caused by the noisy joints is also evenly

distributed over most joints types (with wrist and nose a bit higher). Regarding the actions, the ones with a notably higher joint displacement (above 10cm, greeting, sitting-down, waiting) have a slightly higher average difference, but it seems in general that the errors were well tolerated as the model can use a full sequence of 50 input frames. Most impact on the MPJPE is caused by the action complexity itself.

Action-Class Joint-Type	MPJPE (clean)	MPJPE (input)	above 10cm %	MPJPE (noisy)	Difference (noisy-clean)
Directions	140	54	5	156	16
Discussion	174	53	5	196	22
Eating	90	52	6	103	13
Greeting	235	96	37	262	27
Phoning	101	57	8	124	23
Posing	174	51	6	194	20
Purchases	179	56	8	203	24
Sitting	63	75	15	93	30
SittingDown	99	112	43	120	21
Smoking	106	55	9	125	19
TakingPhoto	227	56	6	244	17
Waiting	164	96	35	196	32
WalkDog	147	48	4	170	23
Walking	225	52	4	255	30
WalkTogether	102	49	6	116	14
Average	143	63	12	165	22
hip_right	104	91	18	125	21
hip_left	103	70	11	124	21
knee_right	111	48	7	131	20
knee_left	109	55	7	126	17
ankle_right	112	81	10	128	16
ankle_left	114	97	13	124	10
nose	146	105	42	189	43
shoulder_right	138	47	8	157	19
shoulder_left	136	43	7	152	16
elbow_right	179	45	8	194	15
elbow_left	175	47	8	189	14
wrist_right	221	45	11	248	27
wrist_left	209	45	9	235	26

Table 14. Errors of *MotionConformer* per action and joint type on the *Human3.6M* dataset using ground-truth or detector-noisy joints.

H. Example of Pseudo-Label Adaptation

Figure 4 shows one representative forecast before and after adaptation on detector-generated pose sequences. While this example is only illustrative, it visualizes the same effect observed quantitatively in Section 8.

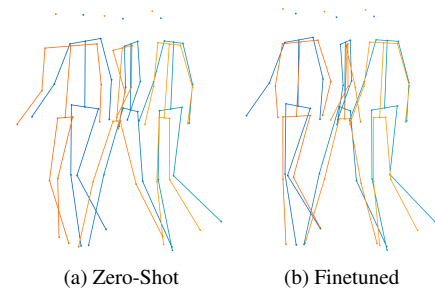


Figure 4. Example of improvements through finetuning on *Human3.6M* dataset. In blue the ground-truth, in orange the prediction. The walking movement was already well continued in (a), but especially the stride distances improved after finetuning. For better visualization some intermediate timesteps are not displayed.