BAL: Bidirectional Autoregression in Latent Space for Learning Human Movement Representations

Genki Kinoshita **Ko Nishino**

Graduate School of Informatics, Kyoto University

gkinoshita@vision.ist.i.kyoto-u.ac.jp

https://vision.ist.i.kyoto-u.ac.jp/research/bal

Abstract

Learning semantically meaningful human movement representations from 3D pose sequences is essential for human behavior modeling. Existing self-supervised methods derive supervision from artificial assumptions, *e.g.*, invariance to augmentation within a certain range or recoverability of masked regions at a certain temporal scale, introducing heuristic hyperparameters that make the learned semantics sensitive to their settings. We propose BAL, a self-supervised framework that instead exploits supervision intrinsic to human movement: the natural ordering structure of everyday behavior. BAL autoregressively predicts in a semantically abstract latent space, analogous to how humans anticipate others' actions rather than their precise joint configurations. Bidirectional autoregressive prediction, both forward and backward in time, fully leverages these ordering constraints while enabling representation extraction using both past and future context at inference. To suppress representation collapse, BAL decomposes each prediction target into a global component capturing sequence-level semantics and a local component capturing context-independent movement. We prove that the local targets retain nontrivial angular diversity unless the representations are completely collapsed, preserving a meaningful training signal. Experiments on large-scale datasets demonstrate that BAL's representations are highly effective for motion captioning, forecasting, and interpolation, outperforming baselines on most evaluation metrics.

1 Introduction

Understanding human movement semantics underpins a broad range of behavior modeling tasks, from motion captioning to forecasting. Among available modalities for movement understanding, 3D pose sequences are particularly attractive: they abstract away appearance and scene context, are inherently privacy-preserving, and are available at a rapidly growing scale. Ideally, movement semantics would be learned from this abundant data using only self-supervision. Yet learning representations that capture these semantics from 3D pose sequences alone remains an open challenge.

Existing self-supervised methods share a fundamental limitation: the supervisory signal is defined by artificial design choices. Contrastive methods [12, 18, 46, 52] require representations to be invariant under hand-crafted augmentations with chosen strengths; masked methods [17, 27, 36, 47, 49, 56] require masked regions to be recoverable from context at a chosen temporal scale. The resulting representations are sensitive to these choices, and obtaining semantics suited to a target task often requires dataset- or task-specific hyperparameter tuning. This sensitivity is not incidental; it follows directly from the fact that the supervision is externally imposed rather than grounded in properties of the data itself.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

A more principled alternative is to exploit supervision intrinsic to human movement. Daily behavior exhibits a consistent ordering structure toward goal completion: one pulls out a chair before sitting, opens a refrigerator before reaching inside, and turns a door handle before walking through. These actions are not independent—they unfold as temporally dependent steps within goal-directed processes. This structure is a natural source of supervision: if representations capture movement semantics, they should reflect these ordering constraints.

The most direct way to exploit this ordering structure is to predict the future from the past. However, frame-level autoregressive prediction in pose space does not naturally produce semantic representations. Short-term joint trajectory smoothness is sufficient to minimize such a loss, so representations tend to encode local pose kinematics rather than action-level semantics. Predicting farther into the future does not resolve this, because the same action can manifest as different poses depending on the person, context, and style; pose-space prediction targets are inherently ambiguous, pushing the model to average over plausible futures rather than distinguish their underlying semantics.

Humans do not reason about others' behavior this way. Rather than estimating future joint positions, we anticipate behavior in semantically abstract units—sitting down, taking something out, leaving the room. Moreover, this semantic prediction operates not only from past to future but also from future to past: knowing someone is reaching into a refrigerator, we naturally infer they opened it moments before. We propose BAL (**B**idirectional **A**utoregression in **L**atent space), a self-supervised framework that turns these observations into a learning objective. BAL divides a pose sequence into fixed-length temporal chunks, compresses each chunk into a single latent token, and trains by predicting these tokens autoregressively in both temporal directions, rather than predicting raw poses. Since latent tokens serve as semantic abstractions of chunk-level movements, ordering constraints among them provide a self-supervisory signal grounded in the structure of behavior itself, reducing the reliance on heuristic hyperparameters to define it. During training, bidirectional prediction more fully exploits the ordering constraints as supervision. Moreover, at inference, it enables the representation extractor to process all chunks without causal masking, extracting chunk representations based on both past and future context in a single forward pass.

A key challenge is that the prediction targets must be learned jointly with their predictors, creating a risk of representation collapse—the extractor degenerates to nearly constant outputs, making prediction trivially easy but semantically vacuous. We address this by decomposing each prediction target into a global component (the mean across all latent tokens in the sequence) and a local component (the residual from that mean). We prove that the L2-normalized local targets always retain nontrivial angular diversity (their maximum pairwise angle is at least $90°$) as long as the latent tokens are not all identical. This means that even as the extractor starts collapsing, predicting normalized local targets does not become trivially easy, helping to preserve a gradient signal that resists further collapse.

We evaluate BAL on large-scale daily activity datasets across three downstream tasks: motion captioning, long-term motion forecasting, and long-term motion interpolation. BAL outperforms the compared self-supervised methods, including masked modeling and contrastive approaches, on most evaluation metrics, suggesting that ordering constraints in latent space provide a more effective supervisory signal for learning movement semantics.

2 Related work

Self-supervised representation learning from 3D poses. A variety of self-supervised objectives have been proposed to learn movement representations from 3D pose sequences [1, 20, 21, 27, 29, 36, 38, 39, 46, 47, 49, 52, 53, 56]. They are broadly grouped into masked modeling [29, 47, 56] and contrastive learning [21, 46]. The former learns by predicting masked input regions, whereas the latter learns by making representations invariant to data augmentations. In both cases, supervision depends on artificial design choices, such as mask size or augmentation type and strength. These choices affect the content of the learned representations, yet are difficult to select and often require task- or dataset-specific tuning. In contrast, BAL uses the ordering structure in daily human behavior as supervision, reducing the reliance on such heuristic design choices.

Most masked-modeling methods [36, 47, 56] compute losses in observation space. Their training can therefore be dominated by minimizing fine-grained joint differences. BAL instead performs

autoregressive prediction in latent space, shifting the focus from reproducing minor pose variations to capturing underlying movement semantics.

A4Mer [17] also defines its objective in latent space using JEPA [2]. However, it requires a complex two-stage training and inference pipeline, along with heuristic hyperparameters, to learn representations at semantically meaningful units. While this is a reasonable direction, we argue that the more critical issue is the choice of self-supervisory signal. Rather than discovering such units with a complex framework, BAL uses the ordering constraints intrinsic to human movement as supervision, enabling simpler, more scalable learning with fewer hyperparameters to define supervision while producing more effective representations.

Self-supervised learning with autoregressive prediction has been successful in various fields, including text [33, 34], speech [35, 43], images [31, 41, 42], and videos [9, 50], while it remains underexplored for human movement. In vision, iGPT [5] and AIM [7] show that representations learned with autoregression achieve higher linear-probe classification accuracy than those learned with masked modeling, suggesting the potential of autoregressive objectives for representation learning.

Autoregressive methods differ in whether they predict in the observation space or in a latent space. A representative example of the latter is CPC [43], which predicts future latent representations from an autoregressively computed context. DPC [13] and wav2vec [35] adapt this idea to RGB videos and speech recognition, respectively. These methods, however, rely on contrastive losses, which require complex objective design, including negative sampling. More recent methods instead directly regress latent representations, as in NEPA [48] for image pretraining and LeWM [26] for world models. BAL follows this direction by minimizing the distance between predicted and target representations.

Still, existing regression-based latent autoregressive methods have notable limitations. NEPA predicts shallow features produced by a single convolutional layer, which remain tied to local input details and lack semantic abstraction; indeed, it reports weak linear-probe performance. LeWM predicts deeper Transformer representations, but requires SIGReg [4] for stable training, introducing additional hyperparameters. Moreover, we find that SIGReg is not sufficient to suppress representation collapse on human movement data. Like LeWM, BAL predicts deep latent representations, but it mitigates representation collapse through a simple decomposition of prediction targets into global and local movement components, without introducing hyperparameters.

Bidirectional autoregression. Since autoregressive prediction is inherently directional, an important challenge is how to combine autoregressive self-supervision with bidirectional contextual inference. In text and speech, several works address this by training separate past-to-future and future-to-past models [8, 32, 33]. Although this exploits both temporal directions, it requires computing two direction-specific representations at inference time, increasing computational cost.

This design largely stems from coupling representation extraction and autoregressive prediction in a single model. Since past-to-future and future-to-past predictions follow different conditional distributions, modeling both directions naturally leads to direction-specific models. BAL instead decouples these two roles by using a shared representation extractor with two direction-specific predictors. The predictors model the direction-specific conditional distributions required for autoregressive prediction, allowing the shared extractor to focus on representing the movement in each corresponding chunk rather than encoding direction-specific predictive states. By using the same extractor to learn past-conditioned and future-conditioned views of each chunk, BAL places these representations in a shared latent space. At inference time, this gives BAL a way to extract representations without causal masks while considering bidirectional context in a single forward pass.

3 BAL

Our goal is to learn movement representations from 3D pose sequences by leveraging the natural ordering constraints inherent in daily human behavior, with minimal dependence on heuristic hyperparameters for defining the supervisory signal. We provide an overview of BAL in Fig. 1.

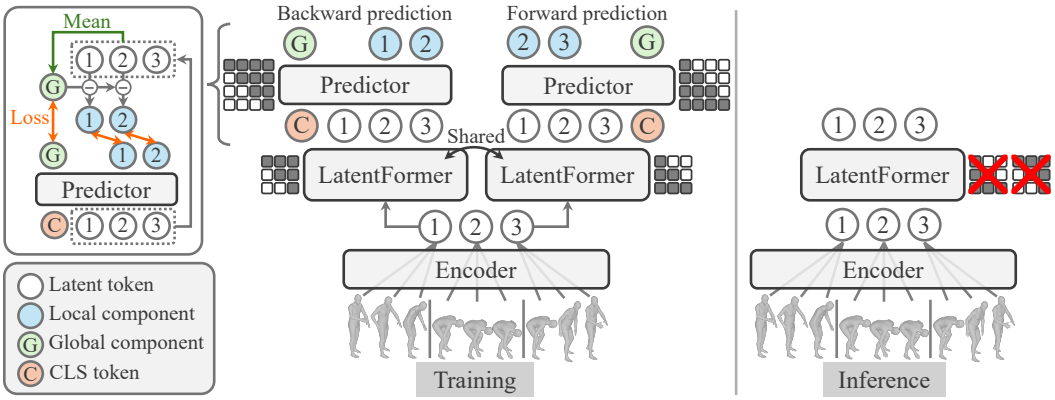


Figure 1: BAL learns movement representations from 3D pose sequences by bidirectional latent-space autoregressive prediction. The Encoder consolidates each chunk into a latent token, and a shared LatentFormer processes these tokens in both directions using causal and reversed causal masks. Two direction-specific predictors predict local components of adjacent latent tokens, while the global component is predicted from a learnable `[CLS]` token. At inference time, the LatentFormer extracts chunk representations without causal masking, enabling bidirectional contextual inference.

3.1 Learning by autoregression in latent space

BAL is trained by autoregressively predicting latent tokens. Each token represents a temporal chunk of movement, reflecting that the ordering structure in daily behavior emerges more naturally between temporally extended units than between individual frames. By predicting chunk-level semantic abstractions rather than fine-grained poses, BAL uses behavioral ordering as semantic supervision.

BAL consists of a representation extractor and a predictor. As in A4Mer [17], the representation extractor comprises an Encoder f_E and a LatentFormer f_L. Given a 3D pose sequence X, BAL divides it into M temporal chunks $X_{1:M}$. For each chunk X_m, the Encoder initializes a chunk embedding by max pooling over the frames in X_m and updates it by alternating self-attention within X_m and cross-attention from the chunk embedding to the frames in X_m:

$$h_m = f_E(X_m)\,, \tag{1}$$

where h_m denotes the resulting chunk embedding. The LatentFormer, a Transformer Encoder [44], models temporal dependencies across chunks and yields a latent token z_m^{fwd} for each chunk, while applying a causal mask to prevent access to future chunks during autoregressive learning:

$$z_m^{\text{fwd}} = f_L(h_{\leq m})\,. \tag{2}$$

Since self-attention in the Encoder is confined to each chunk, the Encoder is effectively limited to local aggregation into chunk embeddings. The LatentFormer therefore operates on already abstracted chunk representations, allowing it to reason about inter-chunk relationships at a semantic level.

The predictor g^{fwd}, a Transformer Encoder with causal masking, then takes $z_{\leq m}^{\text{fwd}}$ and predicts the next latent token:

$$\hat{z}_{m+1}^{\text{fwd}} = g^{\text{fwd}}(z_{\leq m}^{\text{fwd}})\,. \tag{3}$$

We apply normalization to the prediction and the target as appropriate for the chosen objective (see Sec. C.3), and minimize the distance between them:

$$\mathcal{L}^{\text{fwd}} = \frac{1}{M-1}\sum_{m=1}^{M-1} D\left(\hat{z}_{m+1}^{\text{fwd}}, \text{sg}\left(z_{m+1}^{\text{fwd}}\right)\right)\,, \tag{4}$$

where $D(\cdot,\cdot)$ denotes a distance function that incorporates normalization (*e.g.*, L2 normalization or layer normalization [3]), and $\text{sg}(\cdot)$ denotes stop-gradient.

3.2 Bidirectional autoregression and inference

To fully exploit natural ordering constraints as self-supervision while allowing inference that considers both past and future context, BAL learns by latent-space autoregressive prediction in both temporal

directions. The forward and backward directions share the representation extractor but use separate predictors: the LatentFormer applies a standard causal mask in the forward direction and a reversed causal mask in the backward direction, while a dedicated backward predictor g^{bwd} is introduced because past- and future-conditioned predictions follow different conditional distributions. With reversed causal masks in both the LatentFormer and the predictor, the backward direction yields

$$z_m^{\mathrm{bwd}} = f_L(h_{\geq m}), \qquad \hat{z}_{m-1}^{\mathrm{bwd}} = g^{\mathrm{bwd}}(z_{\geq m}^{\mathrm{bwd}})\,. \tag{5}$$

The backward loss is defined analogously to Eq. (4):

$$\mathcal{L}^{\mathrm{bwd}} = \frac{1}{M-1}\sum_{m=2}^{M} D\left(\hat{z}_{m-1}^{\mathrm{bwd}}, \mathrm{sg}\big(z_{m-1}^{\mathrm{bwd}}\big)\right)\,. \tag{6}$$

By sharing the representation extractor and localizing the asymmetry between the two conditional distributions to the predictors, BAL reduces the need for the extracted representations themselves to encode direction-specific predictive states. The shared extractor instead learns past-conditioned and future-conditioned views of each chunk in a common latent space, which gives us a way to remove the causal masks at inference time. As a result, the extractor can produce representations while taking both past and future context into account in a single forward pass. We show in Sec. 4.6 that this bidirectional inference improves downstream performance.

3.3 Decomposition into local and global components

In practice, we find that directly minimizing Eqs. (4) and (6) pushes the representations toward collapse, where the extractor outputs highly similar representations. This arises from a self-reinforcing loop: once the extractor starts producing similar representations, the predictors can reduce the loss simply by outputting near-constant predictions regardless of their inputs, which further drives training toward collapse. Conversely, if predictions from such similar representations remain difficult and incur a sufficiently large loss, collapse can be suppressed. This suggests that preserving sufficient diversity in the prediction targets can suppress collapse.

Based on this, BAL decomposes each target representation into a global component that captures the semantics of the entire sequence and a local component that captures the movement of each chunk independent of the sequence-level semantics. Given latent tokens $z_{1:M}$ of either direction, we define the global component z^g as their mean and the local component z_m^l as the residual:

$$z^g = \frac{1}{M}\sum_{m=1}^{M} z_m\,, \qquad z_m^l = z_m - z^g\,. \tag{7}$$

A key property of this decomposition is that the L2-normalized local components $u_m = z_m^l/\|z_m^l\|$ always retain nontrivial diversity when all local components are nonzero. Formally, writing $a_m = \|z_m^l\| > 0$ and $\mu = \min_{m\neq n} u_m^\top u_n$, the identity $\sum_{m=1}^M z_m^l = 0$ gives $\sum_{m=1}^M a_m u_m = 0$. Taking the squared norm yields

$$0 = \left\|\sum_{m=1}^{M} a_m u_m\right\|^2 = \sum_{m=1}^{M} a_m^2 + 2\sum_{m<n} a_m a_n\, u_m^\top u_n \geq \sum_{m=1}^{M} a_m^2 + 2\mu \sum_{m<n} a_m a_n\,, \tag{8}$$

and therefore

$$\mu \leq -\frac{\sum_{m=1}^M a_m^2}{2\sum_{m<n} a_m a_n}\,. \tag{9}$$

Moreover, Cauchy–Schwarz implies

$$\left(\sum_{m=1}^{M} a_m\right)^2 \leq M\sum_{m=1}^{M} a_m^2 \Longleftrightarrow 2\sum_{m<n} a_m a_n \leq (M-1)\sum_{m=1}^{M} a_m^2\,. \tag{10}$$

Combining these inequalities gives $\mu \leq -1/(M-1)$. Since $\angle(u_m, u_n) = \arccos(u_m^\top u_n)$ and $\arccos$ is decreasing, we obtain

$$\max_{m\neq n} \angle(u_m, u_n) = \arccos\left(\min_{m\neq n} u_m^\top u_n\right) \geq \arccos\left(-\frac{1}{M-1}\right) \geq \frac{\pi}{2}\,. \tag{11}$$

Thus, when all local components are nonzero, at least one pair of normalized local components is separated by an angle of at least 90°. Hence, even when the extractor produces nearly identical representations, the normalized local components remain nontrivially diverse, making them difficult for the predictors to infer from such similar representations; penalizing this prediction error should therefore discourage such degeneracy and suppress collapse. An analogous argument holds when layer normalization is used instead of L2 normalization (see Appendix B).

The predictors in BAL predict the local and global components separately. The forward predictor g^{fwd} predicts z^l_{m+1} from $z^{\mathrm{fwd}}_{\leq m}$, while the backward predictor g^{bwd} predicts z^l_{m-1} from $z^{\mathrm{bwd}}_{\geq m}$. In each direction, the global component is predicted from a learnable `[CLS]` token that attends to the entire sequence without causal masking to summarize the global movement but is not attended to by other tokens. Let $\hat{z}^{l,\mathrm{fwd}}_{m+1}$ and $\hat{z}^{l,\mathrm{bwd}}_{m-1}$ denote the predicted local components in the forward and backward directions, and let $\hat{z}^{g,\mathrm{fwd}}$ and $\hat{z}^{g,\mathrm{bwd}}$ denote the global components predicted from the `[CLS]` token. Using this notation, the local and global losses can be written as

$$\mathcal{L}^{l,\mathrm{fwd}}_m = D\left(\hat{z}^{l,\mathrm{fwd}}_{m+1}, \mathrm{sg}\left(z^l_{m+1}\right)\right),\ \mathcal{L}^{l,\mathrm{bwd}}_m = D\left(\hat{z}^{l,\mathrm{bwd}}_{m-1}, \mathrm{sg}\left(z^l_{m-1}\right)\right), \tag{12}$$

$$\mathcal{L}^{g,\mathrm{fwd}} = D\left(\hat{z}^{g,\mathrm{fwd}}, \mathrm{sg}\left(z^g\right)\right),\ \mathcal{L}^{g,\mathrm{bwd}} = D\left(\hat{z}^{g,\mathrm{bwd}}, \mathrm{sg}\left(z^g\right)\right). \tag{13}$$

The final objective is

$$\mathcal{L} = \frac{1}{M-1}\sum_{m=1}^{M-1}\left(\mathcal{L}^{l,\mathrm{fwd}}_m + \lambda^{\mathrm{fwd}}_m \mathcal{L}^{g,\mathrm{fwd}}\right) + \frac{1}{M-1}\sum_{m=2}^{M}\left(\mathcal{L}^{l,\mathrm{bwd}}_m + \lambda^{\mathrm{bwd}}_m \mathcal{L}^{g,\mathrm{bwd}}\right), \tag{14}$$

where $\lambda^{\mathrm{fwd}}_m = \mathrm{sg}(\mathcal{L}^{l,\mathrm{fwd}}_m / \mathcal{L}^{g,\mathrm{fwd}})$ and $\lambda^{\mathrm{bwd}}_m = \mathrm{sg}(\mathcal{L}^{l,\mathrm{bwd}}_m / \mathcal{L}^{g,\mathrm{bwd}})$ dynamically balance the magnitudes of the local and global losses so that neither term dominates optimization.

A4Mer, which uses JEPA rather than latent-space autoregressive prediction, also decomposes representations into global and local components, but with a different motivation: there, the decomposition serves to learn reusable representations that are independent of context. We reinterpret it as a principled way to suppress representation collapse and provide a theoretical account of how it does so. Moreover, whereas A4Mer decomposes the predicted latent tokens when computing the loss, the predictors in BAL directly predict the global and local components. This clearer separation of the prediction targets makes it easier for the predictors to learn what each output should represent. We empirically find that this leads to better downstream performance than the A4Mer-style variant.

4 Experiments

We evaluate the effectiveness of BAL's representations on three downstream tasks that require understanding movement semantics: motion captioning, forecasting, and interpolation. As baselines, we compare BAL with self-supervised representation learning methods based on different pretext tasks. Each baseline follows its official implementation, including its input pose representation, network architecture, and training hyperparameters. We further conduct a qualitative motion retrieval analysis to inspect whether the learned representation space reflects semantic similarity between motions.

When training task-specific heads for downstream tasks, we freeze the representation extractors for all tasks, since finetuning them could make performance depend more heavily on model size and network architecture than on the quality of the representations learned through the self-supervised objective of each method. We provide architectural details and training settings in Appendix C.

4.1 Datasets

BAL leverages the natural ordering structure in everyday human behavior as self-supervision. We therefore pretrain both BAL and the baselines on a mixture of three datasets that contain natural daily activities: AMD [17], Embody 3D [28], and Nymeria [25].

AMD contains about 14.2 hours of daily household activities recorded in a real house. We follow the official protocol and split the data into training, validation, and test sets with an 8:1:1 ratio.

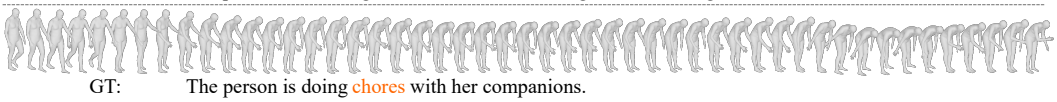
Nymeria contains approximately 300 hours of daily activities recorded in indoor and outdoor environments, including multi-person interactions. We separate sessions by whether they have text

Table 1: Comparison of methods pretrained on AMD, Nymeria (Nym), and Embody 3D (E3D), for motion captioning, forecasting, and interpolation. R-L and BERT denote ROUGE-L and BERTScore, respectively. BAL achieves the best performance on most metrics across tasks.

Method	**Objective**	**Captioning** ↑								**Forecasting** (mm) ↓			**Interpolation** (mm) ↓		
		BLEU-1 [30]		R-L [19]		CIDEr [45]		BERT [54]							
		Nym	E3D	Nym	E3D	Nym	E3D	Nym	E3D	AMD	Nym	E3D	AMD	Nym	E3D
A4Mer [17]	JEPA	15.7	32.9	20.5	22.1	4.3	4.3	8.7	10.2	641.6	359.0	381.3	163.7	145.1	86.3
PUMPS [29]	Reconstruct	8.2	58.7	17.0	44.8	16.2	17.3	9.5	34.3	186.4	453.6	111.7	195.6	472.4	89.2
USDRL [46]	Contrastive	27.1	58.0	34.8	45.3	32.5	18.4	17.7	35.0	136.1	373.8	90.2	117.7	424.6	59.2
BehaveMAE [36]	MAE	33.9	58.8	38.3	44.6	49.4	17.4	24.6	33.6	195.5	394.8	152.7	135.6	103.5	77.8
MotionBERT [56]	Mask & 2D→3D	24.3	58.4	36.6	45.2	19.9	16.7	15.8	35.2	976.6	1623.2	248.7	136.7	348.7	69.5
MacDiff [47]	Mask & Denoise	36.7	58.2	41.4	44.7	65.6	17.8	28.5	33.6	160.0	371.8	73.2	112.7	454.1	56.6
BAL	Latent AR	37.3	58.9	41.2	45.3	63.1	18.6	29.6	35.0	90.5	82.1	63.8	117.5	77.0	59.2



GT: The person is arranging pieces of furniture with a group of people as they make small talk.
BAL: The person is kneeling on the floor, assembling furniture with a group of people.
MacDiff: The person is kneeling on the floor, while talking to their colleagues.



GT: The person is doing chores with her companions.
BAL: The person is actively engaging in a conversation with a group of people while they are doing chores.
USDRL: The person is actively participating in a group activity with their friends.

Figure 2: Motion captioning results. BAL generates more accurate descriptions than the baselines, suggesting that its representations capture the semantics of the movement more effectively.

annotations so that annotated sessions are included in every split for motion captioning. For each subset, we hold out 20% of the sessions as the test set and split the remaining sessions into training and validation sets at an 8:2 ratio.

Embody 3D (E3D) contains about 500 hours of indoor motion sequences. While recorded in a smaller and more constrained environment than the others, it covers diverse behaviors such as charades and multi-person conversations. We split E3D in the same way as Nymeria.

Following A4Mer [17], we downsample input sequences to 5 fps to efficiently handle long sequences while preserving movement semantics. For downstream tasks, we train the forecasting and interpolation heads on the same mixture and evaluate them on each dataset, and train and evaluate the captioning head on E3D and Nymeria, which provide text annotations.

In both pretraining and downstream tasks, the models take input sequences of up to 30 seconds. BAL uses a fixed chunk length of 4 seconds, following the definition of atomic actions in Nymeria as movements lasting 3–5 seconds.

4.2 Motion captioning

To examine whether BAL's representations capture semantics that can be translated into natural language, we perform motion captioning with a large language model (LLM). We project the extracted representations into the LLM text-embedding space using a two-layer MLP, and feed the resulting embeddings into the LLM together with fixed system and user prompts to generate captions.

We use Llama 3.1 8B Instruct [10] as the LLM and keep it frozen, training only the projection MLP to measure pure semantic awareness of the learned representations. Since E3D and Nymeria differ in annotation granularity, we train a separate projection MLP for each dataset.

Tab. 1 reports results on standard text generation metrics. BAL achieves higher scores than the baselines on most metrics. This suggests that its representations better capture movement semantics, allowing the LLM to describe the movement more accurately, as illustrated in Fig. 2.

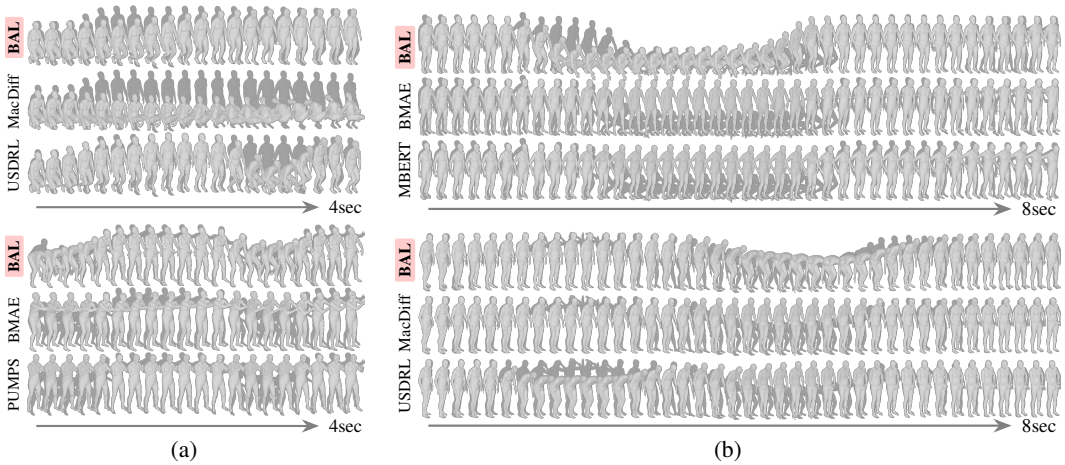


Figure 3: (a) Forecasted poses through autoregressive latent token prediction and (b) interpolated poses with latent token interpolation. The dark-colored SMPL [22] meshes denote the ground truth. BMAE and MBERT denote BehaveMAE and MotionBERT, respectively. The representations learned by BAL enable more accurate and semantically plausible forecasting and interpolation.

4.3 Long-term motion forecasting

If the learned representations capture movement semantics, predicting future representations in latent space should enable forecasting at the semantic level rather than mere smooth extrapolation of poses. To realize such semantically informed motion forecasting, we follow A4Mer and introduce a forecasting head that autoregressively predicts the next latent token from the latent token sequence extracted by the representation extractor. The predicted latent tokens are then mapped back to pose space by a decoder, which is trained independently of the forecasting head. We forecast poses up to 4 seconds ahead and evaluate using MPJPE averaged over all predicted frames.

Tab. 1 shows the results. BAL consistently achieves lower errors than the baselines, suggesting that its representations are more semantically informative and enable the forecasting head to reason about plausible future movement. This is also reflected in the qualitative results in Fig. 3a.

4.4 Long-term motion interpolation

Representations that capture semantics should also help infer missing motion between observed past and future poses. To test this, we follow A4Mer and introduce an interpolation head that predicts latent tokens for the missing interval from those of the observed poses. As in forecasting, the predicted tokens are decoded into poses with the decoder.

For training and evaluation, we randomly mask a 4- to 12-second interval in each input sequence and interpolate the motion within the masked region. We evaluate the interpolated poses using MPJPE averaged over all masked frames.

Tab. 1 shows that BAL achieves lower interpolation errors than most of the baselines, and Fig. 3b provides qualitative examples. These results suggest that BAL's representations enable the interpolation head to infer semantically plausible missing motion from surrounding observations.

In Appendix D.2, we further analyze BAL's large gains in motion forecasting and interpolation by aligning the global translation with the ground truth. This reduces the baselines' errors, suggesting that weaker encoding of global translation partly explains their performance gap with BAL. Yet BAL still outperforms the baselines on most metrics after the alignment.

4.5 Motion retrieval

We perform motion retrieval on the pooled test splits of the three datasets to qualitatively analyze the learned representation space. For each query motion, we retrieve the three nearest neighbors in BAL's representation space, excluding the query itself. As shown in Fig. 4, the retrieved motions are

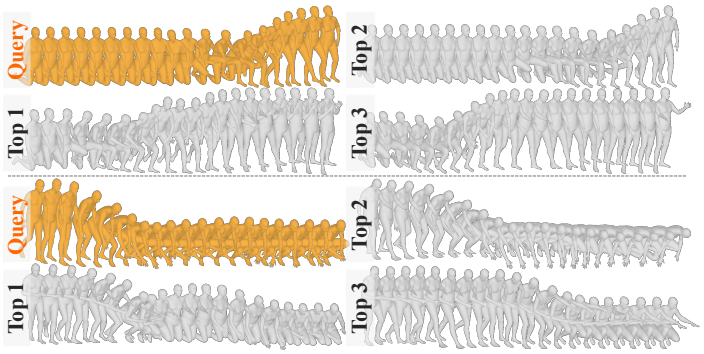


Figure 4: Motion retrieval results. The three nearest neighbors to each query in the representation space correspond to semantically similar movements.

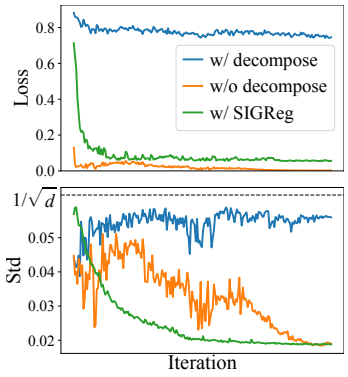


Figure 5: Training loss and feature standard deviation during pretraining.

semantically similar to the queries even when they come from different datasets, suggesting that the learned representation space reflects semantic similarities between motions.

4.6 Ablation study

To assess the effect of key design choices in BAL, we compare variants that remove or replace components of the proposed model in Tab. 2. Pretraining and captioning follow the settings in the previous sections, while the forecasting and interpolation heads are trained for 57,200 iterations to reduce computational cost. All variants use the same architecture for the representation extractor as BAL. The pose-space variant additionally has a Perceiver IO-style decoder [16], and the JEPA [2] variants use predictors with the same architecture as that used in BAL, with prediction targets obtained from an EMA-updated copy of the representation extractor.

Autoregression in latent space. BAL uses the natural ordering of daily human behavior as self-supervision by autoregressively predicting semantically abstract latent tokens. Pose-space prediction instead forces the model to predict detailed future poses, even though the same movement semantics can yield different poses depending on context and individual style. This ambiguity can make the objective less suitable for learning discriminative semantic representations. Indeed, this variant achieves worse performance than BAL across all tasks.

Bidirectional masked prediction methods such as JEPA face a trade-off in mask length. With short masks, past and future contexts complement each other in predicting the target, so neither direction alone needs to retain sufficient information for the prediction. With long masks, positions far from the observed context admit multiple plausible actions as targets. Autoregressive prediction, by contrast, requires a single direction to retain sufficient information to predict the next movement, which is tightly constrained by the observed history. Its prediction errors therefore reflect a lack of information in the representations needed for prediction, rather than uncertainty among plausible actions, providing a more informative supervisory signal. Consistent with this interpretation, BAL outperforms the JEPA variants across all tasks.

Decomposition into local and global components. Following SimSiam [6], we analyze representation collapse by monitoring the training loss and feature standard deviation during pretraining (Fig. 5). For this analysis, we use only 10% of the pretraining steps used in the full experiments to reduce computational cost. The standard deviation is computed over L2-normalized representations and averaged over feature dimensions; it is zero under collapse to a constant and close to $1/\sqrt{d}$ for approximately zero-mean isotropic features in d dimensions. Without our decomposition, or when replacing it with SIGReg [4] as used in LeWM [26], the standard deviation remains low, indicating limited representation diversity and a tendency toward collapse. Correspondingly, the training loss becomes very small in these variants, since such representations make the prediction objective easier to minimize. Our decomposition maintains substantially higher diversity throughout pretraining, mitigating this tendency and leading to consistently better downstream performance. For SIGReg, we use the same number of random projections and regularization weight as Maes *et al*. [26].

Table 2: Ablation studies of key design choices. "A4Mer": A4Mer-style local–global decomposition. "(Bi-)AR": (bidirectional) autoregressive training. "Causal"/"Non-Causal"/"Bi-Causal": inference with a causal mask, without a mask, or with concatenated causal and reverse causal outputs. "Full" corresponds to Bi-AR/Non-Causal. For forecasting, all variants retain the causal mask to prevent target information leakage; thus, the results marked with the same symbol († or ‡) are identical. Motion captioning is evaluated on Nymeria. Parenthesized results report mean (standard deviation) over 10 runs; others are from a single run.

Key	**Substitute**	**Captioning** ↑				**Forecasting** (mm) ↓			**Interpolation** (mm) ↓		
		BLEU-1	R-L	CIDEr	BERT	AMD	Nymeria	E3D	AMD	Nymeria	E3D
Latent AR	Pose-space Bi-AR	17.6	18.9	12.6	7.3	118.6	96.6	64.7	125.7	87.4	60.1
	JEPA, mask 15%	34.5	39.0	46.2	25.2	109.3	107.9	91.6	145.8	112.4	83.4
	JEPA, mask 50%	32.0	38.4	41.7	24.9	102.7	99.0	86.7	133.5	96.8	82.1
Decompose targets	w/o decomp.	33.1	37.9	37.3	22.2	95.5	91.7	75.0	121.6	87.3	64.1
	SIGReg [4, 26]	22.7	33.0	13.0	17.9	297.9	267.7	148.9	155.3	117.2	82.5
	w/o global	27.4	33.4	12.4	16.6	172.9	131.9	104.3	149.0	129.0	84.3
	w/o balancing λ	24.7	28.8	49.6	4.2	170.0	180.2	105.5	157.1	225.7	80.9
	A4Mer [17]	21.7	35.3	24.3	18.6	107.2	92.0	69.8	126.5	89.0	66.2
Bi-AR / Non-Causal	AR/Causal	34.3 (1.5)	39.9 (1.6)	63.7 (1.4)	28.9 (0.8)	89.1 (1.8)†	82.9 (1.8)†	63.6 (0.2)†	121.7 (1.7)	88.3 (8.4)	59.6 (0.2)
	AR/Non-Causal	33.4 (2.0)	38.6 (2.0)	60.3 (1.2)	27.5 (1.3)	89.1 (1.8)†	82.9 (1.8)†	63.6 (0.2)†	119.1 (1.9)	88.0 (4.1)	59.8 (0.2)
	Bi-AR/Causal	35.1 (0.9)	40.8 (0.6)	65.2 (1.5)	29.6 (0.8)	88.3 (1.3)‡	81.3 (1.4)‡	63.2 (0.1)‡	122.7 (2.8)	82.0 (1.6)	59.4 (0.1)
	Bi-AR/Bi-Causal	34.2 (1.2)	39.9 (0.7)	62.6 (1.9)	29.0 (0.7)	88.3 (1.3)‡	81.3 (1.4)‡	63.2 (0.1)‡	124.0 (2.4)	81.6 (2.2)	59.6 (0.1)
Full	—	36.5 (1.2)	41.2 (0.4)	68.3 (1.3)	30.8 (0.8)	88.3 (1.3)‡	81.3 (1.4)‡	63.2 (0.1)‡	121.5 (2.2)	79.0 (1.2)	59.3 (0.1)

Minimizing the loss only on the local components, which capture context-independent movement, degrades performance as it provides no supervision for sequence-level movement semantics. Dynamically balancing the global loss weight prevents optimization from favoring either component and improves performance. Directly predicting the global and local components also outperforms the A4Mer-style variant, which obtains these components by decomposing the predicted latent tokens.

Bidirectional autoregression. BAL uses both past-to-future and future-to-past autoregressive training to fully exploit ordering constraints while supporting bidirectional inference. Bidirectional training improves performance on most metrics (Bi-AR/Causal vs. AR/Causal). Removing the causal mask at inference time further improves performance on captioning and interpolation (Full, *i.e.*, Bi-AR/Non-Causal vs. Bi-AR/Causal); for forecasting, we keep the causal mask in the extractor to prevent future information from leaking into the forecasting head, so the two settings coincide. Mask-free inference even outperforms a two-pass alternative that concatenates representations extracted with causal and reverse causal masks (Full vs. Bi-AR/Bi-Causal). This result indicates that inferring the semantics of each motion chunk jointly from past and future context is more effective than processing the two contexts independently and combining them only after extraction. The benefit from removing the causal masks does not emerge from past-to-future training alone (AR/Non-Causal vs. AR/Causal).

5 Conclusion

We proposed BAL, a self-supervised framework for learning movement representations that capture movement semantics from 3D pose sequences. By autoregressively predicting in latent space, BAL turns the natural ordering structure inherent in everyday human behavior into self-supervision. Bidirectional autoregressive prediction from past to future and from future to past enables BAL to fully exploit this structure and extract representations while taking both past and future context into account. We further analyzed how decomposing prediction targets into global and local components helps prevent representation collapse, and empirically validated its effectiveness. Experiments across diverse downstream tasks show that BAL's representations achieve competitive or superior downstream performance compared with existing self-supervised methods. We hope that BAL will serve as a foundation for human movement representation learning and inspire research in broader domains beyond human movement.

Acknowledgments and Disclosure of Funding

This work was in part supported by JSPS KAKENHI 21H04893 and 26H02534, and JST JPMJAP2305. The authors declare no competing interests.

References

[1] Mohamed Abdelfattah and Alexandre Alahi. S-JEPA: A Joint Embedding Predictive Architecture for Skeletal Action Recognition. In *ECCV*, pages 367–384, 2024.

[2] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-Supervised Learning from Images With a Joint-Embedding Predictive Architecture. In *CVPR*, pages 15619–15629, June 2023.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization, 2016.

[4] Randall Balestriero and Yann LeCun. LeJEPA: Provable and Scalable Self-Supervised Learning Without the Heuristics. *CoRR*, abs/2511.08544, 2025.

[5] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative Pretraining From Pixels. In *Proceedings of the 37th International Conference on Machine Learning*, pages 1691–1703, 2020.

[6] Xinlei Chen and Kaiming He. Exploring Simple Siamese Representation Learning. In *CVPR*, pages 15750–15758, 2021.

[7] Alaaeldin El-Nouby, Michal Klein, Shuangfei Zhai, Miguel Ángel Bautista, Vaishaal Shankar, Alexander T. Toshev, Joshua M. Susskind, and Armand Joulin. Scalable Pre-training of Large Autoregressive Image Models. In *Proceedings of the 41st International Conference on Machine Learning*, pages 12371–12384, 2024.

[8] Ruchao Fan, Amber Afshan, and Abeer Alwan. Bi-APC: Bidirectional Autoregressive Predictive Coding for Unsupervised Pre-Training and Its Application to Children's ASR. In *ICASSP*, pages 7023–7027, 2021. doi: 10.1109/ICASSP39728.2021.9414970.

[9] Songwei Ge, Thomas Hayes, Harry Yang, Xi Yin, Guan Pang, David Jacobs, Jia-Bin Huang, and Devi Parikh. Long Video Generation with Time-Agnostic VQGAN and Time-Sensitive Transformer. *CoRR*, abs/2204.03638, 2022.

[10] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, et al. The Llama 3 Herd of Models. *CoRR*, abs/2407.21783, 2024.

[11] Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating Diverse and Natural 3D Human Motions From Text. In *CVPR*, pages 5152–5161, June 2022.

[12] Tianyu Guo, Hong Liu, Zhan Chen, Mengyuan Liu, Tao Wang, and Runwei Ding. Contrastive Learning from Extremely Augmented Skeleton Sequences for Self-Supervised Action Recognition. In *AAAI*, volume 36, pages 762–770, 2022.

[13] Tengda Han, Weidi Xie, and Andrew Zisserman. Video Representation Learning by Dense Predictive Coding. In *Workshop on Large Scale Holistic Video Understanding, ICCV*, 2019.

[14] Dan Hendrycks and Kevin Gimpel. Gaussian Error Linear Units (GELUs). *CoRR*, abs/1606.08415, 2016.

[15] Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-Key Normalization for Transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4246–4253, 2020. doi: 10.18653/v1/2020.findings-emnlp.379.

[16] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J. Hénaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver IO: A General Architecture for Structured Inputs & Outputs. In *ICLR*, 2022.

[17] Genki Kinoshita, Shu Nakamura, Ryo Kawahara, Shohei Nobuhara, Yasutomo Kawanishi, and Ko Nishino. Action Motifs: Self-Supervised Hierarchical Representation of Human Body Movements. In *CVPR*, 2026.

[18] Linguo Li, Minsi Wang, Bingbing Ni, Hang Wang, Jiancheng Yang, and Wenjun Zhang. 3D Human Action Representation Learning via Cross-View Consistency Pursuit. In *CVPR*, pages 4741–4750, June 2021.

[19] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, 2004.

[20] Lilang Lin, Sijie Song, Wenhan Yang, and Jiaying Liu. MS2L: Multi-Task Self-Supervised Learning for Skeleton Based Action Recognition. In *ACM MM*, pages 2490–2498, 2020.

[21] Lilang Lin, Jiahang Zhang, and Jiaying Liu. Actionlet-Dependent Contrastive Learning for Unsupervised Skeleton-Based Action Recognition. In *CVPR*, pages 2363–2372, 2023.

[22] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A Skinned Multi-Person Linear Model. *ACM TOG*, 34(6):248:1–248:16, 2015.

[23] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.

[24] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2019.

[25] Lingni Ma, Yuting Ye, Fangzhou Hong, Vladimir Guzov, Yifeng Jiang, Rowan Postyeni, Luis Pesqueira, Alexander Gamino, Vijay Baiyya, Hyo Jin Kim, et al. Nymeria: A Massive Collection of Multimodal Egocentric Daily Motion in the Wild. In *ECCV*, pages 445–465, 2024.

[26] Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. LeWorldModel: Stable End-to-End Joint-Embedding Predictive Architecture from Pixels. *CoRR*, abs/2603.19312, 2026.

[27] Yunyao Mao, Jiajun Deng, Wengang Zhou, Yao Fang, Wanli Ouyang, and Houqiang Li. Masked Motion Predictors Are Strong 3D Action Representation Learners. In *ICCV*, pages 10181–10191, 2023.

[28] Claire McLean, Makenzie Meendering, Tristan Swartz, Orri Gabbay, Alexandra Olsen, Rachel Jacobs, Nicholas Rosen, Philippe de Bree, Tony Garcia, Gadsden Merrill, Jake Sandakly, Julia Buffalini, Neham Jain, Steven Krenn, Moneish Kumar, Dejan Markovic, Evonne Ng, Fabian Prada, Andrew Saba, Siwei Zhang, Vasu Agrawal, Tim Godisart, Alexander Richard, and Michael Zollhoefer. Embody 3D: A Large-scale Multimodal Motion and Behavior Dataset. Technical report, arXiv, 2025.

[29] Clinton Ansun Mo, Kun Hu, Chengjiang Long, Dong Yuan, Wan-Chi Siu, and Zhiyong Wang. PUMPS: Skeleton-Agnostic Point-Based Universal Motion Pre-Training for Synthesis in Human Motion Tasks. In *ICCV*, pages 14496–14506, 2025.

[30] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A method for automatic evaluation of machine translation. In *ACL*, pages 311–318, 2002. doi: 10.3115/1073083.1073135.

[31] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image Transformer. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4055–4064, 2018.

[32] Matthew E. Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised Sequence Tagging with Bidirectional Language Models. In *ACL*, pages 1756–1765, 2017.

[33] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237, 2018.

[34] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training. Technical report, OpenAI, 2018.

[35] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised Pre-Training for Speech Recognition. In *Interspeech 2019*, pages 3465–3469, 2019. doi: 10.21437/Interspeech.2019-1873.

[36] Lucas Stoffl, Andy Bonnetto, Stéphane d'Ascoli, and Alexander Mathis. Elucidating the Hierarchical Nature of Behavior with Masked Autoencoders. In *ECCV*, pages 106–125, 2024.

[37] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063, 2024.

[38] Yukun Su, Guosheng Lin, and Qingyao Wu. Self-Supervised 3D Skeleton Action Representation Learning with Motion Consistency and Continuity. In *ICCV*, pages 13328–13338, 2021.

[39] Shengkai Sun, Zefan Zhang, Jianfeng Dong, Zhiyong Cheng, Xiaojun Chang, and Meng Wang. Towards Efficient General Feature Prediction in Masked Skeleton Modeling. In *ICCV*, pages 12212–12221, 2025.

[40] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going Deeper with Image Transformers. In *ICCV*, pages 32–42, 2021.

[41] Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel Recurrent Neural Networks. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1747–1756, 2016.

[42] Aäron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. Conditional Image Generation with PixelCNN Decoders. In *NeurIPS*, volume 29, 2016.

[43] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding. *CoRR*, abs/1807.03748, 2018.

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *NeurIPS*, volume 30, 2017.

[45] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. CIDEr: Consensus-based image description evaluation. In *CVPR*, pages 4566–4575, 2015.

[46] Wanjiang Weng, Hongsong Wang, Junbo Wang, Lei He, and Guo-Sen Xie. USDRL: Unified Skeleton-Based Dense Representation Learning with Multi-Grained Feature Decorrelation. In *AAAI*, volume 39, pages 8332–8340, 2025.

[47] Lehong Wu, Lilang Lin, Jiahang Zhang, Yiyang Ma, and Jiaying Liu. MacDiff: Unified Skeleton Modeling with Masked Conditional Diffusion. In *ECCV*, pages 110–128, 2024.

[48] Sihan Xu, Ziqiao Ma, Wenhao Chai, Xuweiyi Chen, Weiyang Jin, Joyce Chai, Saining Xie, and Stella X. Yu. Next-Embedding Prediction Makes Strong Vision Learners. *CoRR*, abs/2512.16922, 2025.

[49] Hong Yan, Yang Liu, Yushen Wei, Zhen Li, Guanbin Li, and Liang Lin. SkeletonMAE: Graph-Based Masked Autoencoder for Skeleton Sequence Pre-Training. In *ICCV*, pages 5606–5618, 2023.

[50] Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. VideoGPT: Video generation using VQ-VAE and transformers. *CoRR*, abs/2104.10157, 2021.

[51] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. XLNet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, volume 32, pages 5754–5764, 2019.

[52] Jiahang Zhang, Lilang Lin, and Jiaying Liu. Hierarchical Consistent Contrastive Learning for Skeleton-Based Action Recognition with Growing Augmentations. In *AAAI*, volume 37, pages 3427–3435, 2023.

[53] Jiahang Zhang, Lilang Lin, and Jiaying Liu. Prompted Contrast with Masked Motion Modeling: Towards Versatile 3D Action Representation Learning. In *ACM MM*, pages 7175–7183, 2023.

[54] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. In *ICLR*, 2020.

[55] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the Continuity of Rotation Representations in Neural Networks. In *CVPR*, pages 5745–5753, 2019.

[56] Wentao Zhu, Xiaoxuan Ma, Zhaoyang Liu, Libin Liu, Wayne Wu, and Yizhou Wang. MotionBERT: A Unified Perspective on Learning Human Motion Representations. In *ICCV*, pages 15085–15099, 2023.

A Algorithm of BAL

Alg. 1 presents PyTorch-style pseudocode for BAL pretraining.

B Diversity of normalized local prediction targets

In Sec. 3.3, we established an angular separation bound for L2-normalized local components when all local components are nonzero. Here, we extend this result to zero local components and derive a corresponding bound for layer normalization [3], including the effect of numerical stabilization. For $M \geq 2$ latent tokens in $\mathbb{R}^d$ with $d \geq 2$, the local components satisfy

$$z_m^l = z_m - \frac{1}{M}\sum_{n=1}^{M} z_n\,, \qquad \sum_{m=1}^{M} z_m^l = 0\,. \tag{15}$$

Algorithm 1 PyTorch-style pseudocode for BAL pretraining

```
# f_E: Encoder
# f_L: LatentFormer
# g_f, g_b: forward and backward predictors
# D: distance function, including normalization
# B: batch size; T: number of frames; M: number of chunks
# d_in: input pose dimension; d: latent dimension

for X in loader: # [B, T, d_in]
    optimizer.zero_grad()

    h = f_E(X) # [B, M, d]

    z_f = f_L(h, mask="causal") # [B, M, d]
    z_b = f_L(h, mask="reverse_causal") # [B, M, d]

    z_l_f, z_g_f = decompose(z_f) # [B, M, d], [B, d]
    z_l_b, z_g_b = decompose(z_b) # [B, M, d], [B, d]

    # global components are predicted from a learnable CLS token
    pred_l_f, pred_g_f = g_f(z_f) # [B, M, d], [B, d]
    pred_l_b, pred_g_b = g_b(z_b) # [B, M, d], [B, d]

    # local loss
    loss_l_f = D(pred_l_f[:, 0:M-1, :], z_l_f.detach()[:, 1:M, :]) # [B, M-1]
    loss_l_b = D(pred_l_b[:, 1:M, :], z_l_b.detach()[:, 0:M-1, :]) # [B, M-1]

    # global loss
    loss_g_f = D(pred_g_f, z_g_f.detach()).unsqueeze(-1) # [B, 1]
    loss_g_b = D(pred_g_b, z_g_b.detach()).unsqueeze(-1) # [B, 1]

    lambda_f = (loss_l_f / loss_g_f).detach() # [B, M-1]
    lambda_b = (loss_l_b / loss_g_b).detach() # [B, M-1]

    loss_f = (loss_l_f + lambda_f * loss_g_f).mean()
    loss_b = (loss_l_b + lambda_b * loss_g_b).mean()
    loss = loss_f + loss_b

    loss.backward() # back-propagate
    optimizer.step() # update parameters

def decompose(z):
    z_global = z.mean(dim=1) # [B, d]
    z_local = z - z_global[:, None, :] # [B, M, d]
    return z_local, z_global

def D(pred, target, is_l2_norm=False):
    if is_l2_norm:
        pred = normalize(pred, dim=-1)
        target = normalize(target, dim=-1)
        loss = 1 - (pred * target).sum(dim=-1)

    else:
        # pred is left unnormalized
        target = layer_norm(target, dim=-1)
        loss = smooth_l1(pred, target, reduction="none").mean(dim=-1)

    return loss
```

B.1 L2 normalization with zero local components

Suppose the latent tokens are not all identical, and let K denote the number of nonzero local components. The zero-sum property in Eq. (15) implies $K \geq 2$, since a single nonzero component cannot sum to zero. Without loss of generality, let these be the first K components, and define

$$a_m = \|z_m^l\| > 0\,, \qquad u_m = \frac{z_m^l}{a_m}\,. \tag{16}$$

Since $z_m^l = a_m u_m$, Eq. (15) gives

$$\sum_{m=1}^{K} a_m u_m = \sum_{m=1}^{M} z_m^l = 0\,. \tag{17}$$

Let $\mu = \min_{1 \le m < n \le K} u_m^\top u_n$. Taking the squared norm of Eq. (17) yields

$$\begin{aligned} 0 &= \left\| \sum_{m=1}^{K} a_m u_m \right\|^2 \\ &= \sum_{m=1}^{K} a_m^2 + 2 \sum_{m<n} a_m a_n \, u_m^\top u_n \\ &\ge \sum_{m=1}^{K} a_m^2 + 2\mu \sum_{m<n} a_m a_n \,. \end{aligned} \tag{18}$$

Cauchy–Schwarz also gives

$$\begin{aligned} 2 \sum_{m<n} a_m a_n &= \left(\sum_{m=1}^{K} a_m \right)^2 - \sum_{m=1}^{K} a_m^2 \\ &\le (K-1) \sum_{m=1}^{K} a_m^2 \,. \end{aligned} \tag{19}$$

Combining these inequalities, we obtain

$$\mu \le -\frac{\sum_{m=1}^{K} a_m^2}{2 \sum_{m<n} a_m a_n} \le -\frac{1}{K-1} \,. \tag{20}$$

As $\arccos$ is decreasing,

$$\max_{m<n} \angle(u_m, u_n) \ge \arccos\left(-\frac{1}{K-1} \right) > \frac{\pi}{2} \,. \tag{21}$$

Thus, the nonzero L2-normalized local components necessarily contain a pair separated by more than 90°. Zero local components are excluded because their L2 normalization is undefined. For $K = M$, this recovers the result in Sec. 3.3.

B.2 Layer normalization

We consider layer normalization without affine parameters, as used for the prediction targets to which stop-gradient is applied. Let P denote the centering matrix that subtracts the mean across feature dimensions:

$$\mathbf{1} = (1, \dots, 1)^\top \in \mathbb{R}^d \,, \qquad P = I_d - \frac{1}{d} \mathbf{1}\mathbf{1}^\top \,. \tag{22}$$

We denote layer normalization with a numerical-stability constant $\epsilon \ge 0$ by LN_ϵ, defined as

$$\mathrm{LN}_\epsilon(x) = \frac{Px}{\sqrt{\|Px\|^2/d + \epsilon}} \,. \tag{23}$$

For $\epsilon = 0$, this operation is defined only when $Px \neq 0$. By linearity of P, Eq. (15) implies

$$\sum_{m=1}^{M} P z_m^l = P \left(\sum_{m=1}^{M} z_m^l \right) = 0 \,. \tag{24}$$

Let K now denote the number of nonzero feature-centered local components. If $K = 0$, all outputs are zero for $\epsilon > 0$ and undefined for $\epsilon = 0$. This can occur even when the latent tokens are not identical, since P removes components that are constant across feature dimensions. If $K > 0$, the zero-sum property in Eq. (24) implies $K \ge 2$.

Without loss of generality, let the first K feature-centered components be nonzero, and define

$$b_m = \|P z_m^l\| > 0 \,, \qquad w_m = \frac{P z_m^l}{b_m} \,, \qquad v_m = \frac{b_m^2}{d} \,. \tag{25}$$

Here, v_m is the variance of z_m^l across feature dimensions. Since

$$\sum_{m=1}^{K} b_m w_m = \sum_{m=1}^{M} P z_m^l = 0\,, \tag{26}$$

the argument in the preceding subsection applies to these feature-centered components, yielding

$$\min_{m<n} w_m^\top w_n \le -\frac{1}{K-1}\,. \tag{27}$$

From Eq. (23), the normalized outputs can be written as

$$\mathrm{LN}_\epsilon(z_m^l) = r_m w_m\,, \qquad r_m = \sqrt{\frac{d v_m}{v_m + \epsilon}}\,. \tag{28}$$

Let $\nu = \min_{1\le m\le K} v_m > 0$. Then

$$r_m \ge r_{\min} = \sqrt{\frac{d\nu}{\nu+\epsilon}}\,. \tag{29}$$

For a pair $m < n$ satisfying Eq. (27), we obtain

$$\begin{aligned} \|r_m w_m - r_n w_n\|^2 &= r_m^2 + r_n^2 - 2 r_m r_n w_m^\top w_n \\ &\ge r_m^2 + r_n^2 + \frac{2 r_m r_n}{K-1} \\ &\ge \frac{2dK}{K-1}\,\frac{\nu}{\nu+\epsilon}\,. \end{aligned} \tag{30}$$

Consequently, the maximum pairwise distance among the normalized nonzero feature-centered local components satisfies

$$\begin{aligned} D_\epsilon &:= \max_{1\le m<n\le K} \|r_m w_m - r_n w_n\| \\ &\ge \sqrt{\frac{2dK}{K-1}\,\frac{\nu}{\nu+\epsilon}}\,. \end{aligned} \tag{31}$$

To interpret this bound, consider first $\epsilon = 0$, for which every defined layer-normalized local component has norm $\sqrt{d}$ by Eq. (28). These components lie in the zero-mean feature subspace, where the maximum possible distance between two vectors of this norm is $2\sqrt{d}$, attained by a pair of opposite vectors. Dividing Eq. (31) by this maximum gives

$$\frac{D_0}{2\sqrt{d}} \ge \sqrt{\frac{K}{2(K-1)}} \ge \frac{1}{\sqrt{2}} \approx 0.707\,. \tag{32}$$

Thus, when $\epsilon = 0$, at least one pair of layer-normalized local targets is separated by at least approximately 70.7% of the maximum possible distance. This lower bound is independent of both the number of tokens and the magnitudes of their nonzero feature-centered local components. In particular, the separation persists as the latent tokens approach one another, provided these components remain nonzero.

For $\epsilon > 0$, Eq. (31) gives the corresponding bound with the factor $\sqrt{\nu/(\nu+\epsilon)}$. The separation therefore remains close to the $\epsilon = 0$ bound when the nonzero feature-wise variances are large relative to the stability constant. Together, these results show that local–global decomposition maintains nontrivial diversity among normalized local prediction targets even when the underlying representations are similar. Near-constant predictions cannot simultaneously match such separated targets, helping to counteract the self-reinforcing collapse mechanism described in Sec. 3.3.

C Implementation details

C.1 Text preprocessing for motion captioning

Nymeria [25] provides text annotations at three levels of granularity. We use the coarsest annotation level, "activity summarization," which describes high-level activities rather than fine-grained motion details. Since these annotations refer to the actor as "C," we replace "C" with "the person" to make the text more natural and easier for the LLM to process.

Embody 3D (E3D) [28] also provides text annotations at multiple levels of granularity. As with Nymeria, we use the coarsest annotation level, "describe person action."

Table 3: Network architectures. All Transformer blocks use layer normalization (LN) [3], width 256, 4 attention heads, MLP width 512, GELU activations [14], QK normalization [15] with a learnable scale and no bias, RoPE [37] with $\theta = 1000$, and LayerScale [40]. For the head modules listed in the Module column, #Layers denotes the number of linear layers.

Model	Module	Token Dim.	#Layers	Component
BAL	Encoder	256	8	Transformer blocks
	LatentFormer	256	8	Transformer blocks
	Predictor	256	3	Transformer blocks
	Predictor head	256	2	LN + linear layers
Decoder	Decoder model	256	4	Transformer blocks
	Head	256	1	LN + linear layer
Forecasting head	Forecasting model	256	4	Transformer blocks
	Head	256	1	LN + linear layer
Interpolation head	Interpolation model	256	4	Transformer blocks
	Head	256	1	LN + linear layer
Captioning	MLP	$256 \to 2176 \to 4096$	2	Linear + GELU + linear

C.2 Pose representation

Following HumanML3D [11], we use an input pose representation in which body shape is normalized and each pose is expressed in a gravity-aligned local coordinate frame. Each pose is represented by $(\boldsymbol{c}, \dot{r}^a, \dot{\boldsymbol{r}}^v, \boldsymbol{j}, \boldsymbol{j}^v, \boldsymbol{j}^r)$, where $\boldsymbol{c} \in \{0,1\}^4$ indicates foot-ground contacts for the left and right heels and toes, $\dot{r}^a \in \mathbb{R}$ denotes the pelvis angular velocity around the gravity axis, and $\dot{\boldsymbol{r}}^v \in \mathbb{R}^2$ denotes the pelvis velocity on the ground plane. The terms $\boldsymbol{j} \in \mathbb{R}^{J\times 3}$, $\boldsymbol{j}^v \in \mathbb{R}^{J\times 3}$, and $\boldsymbol{j}^r \in \mathbb{R}^{J\times 6}$ correspond to the joint positions, joint velocities, and joint rotations, respectively, with J denoting the number of joints. We use the continuous 6D rotation representation [55] for $\boldsymbol{j}^r$. For baseline methods, we use the pose representation of each original paper so that each baseline is evaluated with the representation it was designed for.

C.3 Network architecture

We summarize the network architectures of BAL, the decoder, the forecasting head, the interpolation head, and the MLP for captioning in Tab. 3. The task-specific modules share the same architecture across all methods. Since the representation dimensionality of the baselines varies and differs from that of BAL, we use a learnable linear layer to project each baseline representation to 256 dimensions before the decoder, forecasting head, and interpolation head. For captioning, we instead adjust the input dimension of the two-layer MLP to match each representation, while keeping its hidden and output dimensions unchanged.

Normalization and distance function. For the distance function in Eqs. (4), (6), (12), and (13), we use layer normalization and the smooth L1 distance, rather than L2 normalization and cosine dissimilarity, as this choice yields better downstream performance (Tab. 4). With L2 normalization, both the prediction and the target are normalized. With layer normalization, in contrast, we normalize only the target and leave the predictor outputs unnormalized for distance computation, allowing the final linear layer of the predictor to learn the affine correction needed to match the target distribution.

One possible explanation for the performance gap lies in how the two objectives constrain the predictor outputs. With L2 normalization and cosine dissimilarity, the loss is invariant to positive rescaling of the predictor output, leaving its magnitude unconstrained. In contrast, the smooth L1 distance directly penalizes coordinate-wise deviations of the unnormalized predictor output from the layer-normalized target. For a fixed target, the objective is uniquely minimized when the predictor output matches the target, thus constraining its magnitude as well as its direction.

Table 4: Comparison of normalization operations and distance functions across motion captioning, forecasting, and interpolation. Layer normalization with the smooth L1 distance yields better downstream performance than L2 normalization with cosine dissimilarity across all tasks.

Normalization	**Distance**	**Captioning** ↑								**Forecasting** (mm) ↓			**Interpolation** (mm) ↓		
		BLEU-1		ROUGE-L		CIDEr		BERTScore							
		Nym	E3D	Nym	E3D	Nym	E3D	Nym	E3D	AMD	Nym	E3D	AMD	Nym	E3D
L2 norm	Cosine	28.9	58.3	39.4	44.7	50.8	17.1	27.2	33.9	96.7	84.6	68.8	121.8	81.7	63.1
Layer norm	Smooth L1	37.3	58.9	41.2	45.3	63.1	18.6	29.6	35.0	90.5	82.1	63.8	117.5	77.0	59.2

Table 5: Training settings. All modules listed in the table are optimized with AdamW [24] and gradient norm clipping at 1.0, using linear warmup followed by a cosine learning-rate schedule [23]. For BAL, the weight decay is increased from 0.04 to 0.4 following a cosine schedule. All modules are trained on a single NVIDIA A100 GPU.

Task	**Batch size**	**Learning rate**	**Weight decay**	**Iterations**	**Warmup iter.**	**Training time (h)**
BAL	128	5×10^{-4}	$0.04 \rightarrow 0.4$	429K	2000	39.5
Decoder	128	1×10^{-3}	0.01	143K	2000	4.9
Forecasting	128	1×10^{-3}	0.01	286K	2000	9.6
Interpolation	128	1×10^{-3}	0.01	143K	2000	21.5
Captioning	64	8×10^{-5}	0.0	2.6K	1000	2.3

C.4 Training settings

We summarize the training settings for BAL pretraining and each downstream task in Tab. 5. In all downstream tasks, the pretrained representation extractors, including those of the baselines, are kept frozen, and only the task-specific modules are trained. These task-specific modules are trained on the same dataset and with the same settings for all methods, so that the downstream results reflect the quality of the learned representations rather than differences in training data or configurations.

Motion captioning. For motion captioning, we project the movement representations extracted by the representation extractor into the language embedding space of an LLM using a two-layer MLP. We then construct the input to the LLM by concatenating the system-prompt embeddings, the projected movement embeddings, and the user-prompt embeddings in this order. We use the following system prompt: "You are an assistant that describes human motion concisely in natural English. Describe only what is observable from the given motion." For the user prompt, we use: "Describe this human motion sequence in natural English as a short paragraph in chronological order."

We train only the projection MLP with the standard autoregressive negative log-likelihood loss, keeping the representation extractor and the LLM frozen. The loss is computed only over the ground-truth caption tokens; the system prompt, user prompt, and projected movement embeddings are excluded.

Decoder. Since motion forecasting and interpolation are performed in the latent space, we introduce a decoder that maps latent tokens to poses so that the predictions can be evaluated in pose space. The decoder is implemented as a simple Transformer Encoder [44]. Its input is obtained by repeating each latent token over the frames of the chunk it represents, yielding a frame-level sequence of length T; the decoder then maps this sequence to the corresponding poses. Following A4Mer [17], we train

the decoder with a reconstruction objective $\mathcal{L}_{\text{dec}}$ that combines joint-position and velocity losses:

$$\mathcal{L}_{\text{dec}} = \lambda_{\text{pos}}\mathcal{L}_{\text{pos}} + \lambda_{\text{vel}}\mathcal{L}_{\text{vel}} \,, \tag{33}$$

$$\mathcal{L}_{\text{pos}} = \frac{1}{JT}\sum_{t=1}^{T}\sum_{j=1}^{J}\left\|\hat{\boldsymbol{j}}_{j,t} - \boldsymbol{j}_{j,t}\right\| \,, \tag{34}$$

$$\mathcal{L}_{\text{vel}} = \frac{1}{J(T-1)}\sum_{t=2}^{T}\sum_{j=1}^{J}\left\|\hat{\Delta\boldsymbol{j}}_{j,t} - \Delta\boldsymbol{j}_{j,t}\right\| \,, \tag{35}$$

where $\boldsymbol{j}_{j,t}$ and $\hat{\boldsymbol{j}}_{j,t}$ denote the ground-truth and reconstructed positions of joint j at frame t, respectively. The velocities are defined as $\Delta\boldsymbol{j}_{j,t} = \boldsymbol{j}_{j,t} - \boldsymbol{j}_{j,t-1}$ and $\hat{\Delta\boldsymbol{j}}_{j,t} = \hat{\boldsymbol{j}}_{j,t} - \hat{\boldsymbol{j}}_{j,t-1}$. We set $\lambda_{\text{pos}} = 1.5$ and $\lambda_{\text{vel}} = 1.0$.

Motion forecasting is performed by predicting the next latent token z^{fwd}_{m+1} from the latent token sequence $z^{\text{fwd}}_{\leq m}$ produced by the pretrained representation extractor, and then decoding the predicted token into pose space. Following A4Mer, we train the autoregressive forecasting head using a latent-token forecasting loss $\mathcal{L}_{\text{token}}$ and a joint-space reconstruction loss $\mathcal{L}_{\text{dec}}$ computed by decoding the predicted latent tokens with the frozen decoder:

$$\mathcal{L}_{\text{fore}} = \mathcal{L}_{\text{token}} + \mathcal{L}_{\text{dec}} \,, \tag{36}$$

$$\mathcal{L}_{\text{token}} = \frac{1}{M-1}\sum_{m=1}^{M-1}\text{SmoothL1}\left(\hat{z}^{\text{fwd}}_{m+1} - z^{\text{fwd}}_{m+1}\right) \,, \tag{37}$$

where $\hat{z}^{\text{fwd}}_{m+1}$ denotes the latent token predicted by the head. The reconstruction term helps prevent the forecasting head from optimizing latent dimensions that are not relevant to reconstructing plausible movement.

When A4Mer is used as the representation extractor, its latent tokens correspond to variable-length chunks. Since the lengths of future chunks are unavailable at inference time, we follow the original A4Mer protocol and train the forecasting head to estimate the next chunk length as well. Because the distribution of chunk lengths has a long right tail, we regress them in log space:

$$\mathcal{L}_{\text{len}} = \frac{1}{M-1}\sum_{m=1}^{M-1}\left|\log l_{m+1} - \text{Softplus}\left(\hat{l}_{m+1}\right)\right| \,, \tag{38}$$

where $\hat{l}_{m+1}$ denotes the raw output of the head and l_{m+1} denotes the chunk length determined by the pretrained A4Mer model rather than by manual annotation. In this case, the final forecasting loss is

$$\mathcal{L}_{\text{fore}} = \mathcal{L}_{\text{token}} + \mathcal{L}_{\text{dec}} + \mathcal{L}_{\text{len}} \,. \tag{39}$$

During training, the predicted latent tokens are decoded with the ground-truth chunk lengths. At inference time, we recover the estimated chunk lengths by exponentiating the predicted log lengths and use them for decoding.

Motion interpolation is performed by predicting the latent tokens corresponding to unobserved frames of the input sequence and decoding the completed latent token sequence into pose space. We first extract latent tokens from the masked input sequence, in which the masked frames are excluded from attention through an attention mask. We then replace the latent tokens at the positions corresponding to the masked regions with learnable tokens and feed the resulting sequence to an interpolation head.

For methods with fixed-length chunks, the latent positions associated with the masked regions are determined directly from the temporal grid. In contrast, A4Mer produces latent tokens from an adaptive segmentation, so its tokens correspond to variable-length chunks. A strict interpolation setup for A4Mer would therefore require inferring the adaptive chunk partition within each masked region, including both the number of missing chunks and their lengths. Since this introduces a nontrivial extension of A4Mer, we instead determine the chunk partition by applying the pretrained A4Mer extractor to the fully observed sequence. Note that this gives A4Mer access to the chunk partition of the unobserved frames, which favors A4Mer in this comparison.

Table 6: Comparison with an XLNet-style variant. BAL outperforms it on most metrics.

Method	Captioning ↑								Forecasting (mm) ↓			Interpolation (mm) ↓		
	BLEU-1		ROUGE-L		CIDEr		BERTScore							
	Nym	E3D	Nym	E3D	Nym	E3D	Nym	E3D	AMD	Nym	E3D	AMD	Nym	E3D
XLNet	35.2	57.9	38.6	43.5	42.9	16.3	26.7	32.3	85.8	87.1	65.6	116.5	79.3	58.1
BAL	37.3	58.9	41.2	45.3	63.1	18.6	29.6	35.0	90.5	82.1	63.8	117.5	77.0	59.2

We train the interpolation head with a latent reconstruction loss $\mathcal{L}_{\text{latent}}$ between the predicted latent tokens $\hat{z}_m$ and the target latent tokens z_m extracted from the fully observed sequence by the representation extractor without causal masking:

$$\mathcal{L}_{\text{latent}} = \frac{1}{M} \sum_{m=1}^{M} \text{SmoothL1} \left(\hat{z}_m - z_m \right) . \tag{40}$$

Although the main objective is to recover the latent tokens of the masked regions, we compute the loss over all latent positions to account for the distributional discrepancy between latent tokens extracted from masked and fully observed inputs. As in motion forecasting, we further add the joint-space reconstruction loss $\mathcal{L}_{\text{dec}}$ by decoding the completed latent token sequence with the frozen decoder. The final interpolation loss is

$$\mathcal{L}_{\text{interp}} = \mathcal{L}_{\text{latent}} + \mathcal{L}_{\text{dec}} . \tag{41}$$

D Additional experiments

D.1 XLNet

To assess whether bidirectional autoregressive prediction along the natural temporal order provides more effective self-supervision than prediction over random factorization orders, as in XLNet [51], we compare BAL with a model trained using an XLNet-style objective.

To isolate the effect of the prediction objective as much as possible, we retain BAL's Encoder for producing chunk embeddings and replace the LatentFormer and predictors with XLNet-style two-stream attention comprising a content stream and a query stream. The two streams share a single set of parameters, and the resulting network has the same depth, width, and number of attention heads as the LatentFormer. All other training settings are also identical to those of BAL. In this network, the content stream encodes both the context and each token itself as in standard self-attention, whereas the query stream encodes the target position and aggregates information from the content stream of the tokens preceding the target in the factorization order, without accessing the content of the target token. The final query-stream output is passed to the same predictor head as in BAL (see Tab. 3) to predict the final content-stream representation of the target token. The target is detached from the computational graph, and the loss is computed in latent space as in BAL. Following XLNet, approximately one sixth of the latent tokens in each sequence are selected as prediction targets in the form of contiguous spans of at most five tokens. At inference time, we discard the query stream and extract representations from the content stream without permutation masking.

As shown in Tab. 6, BAL outperforms the XLNet-style model, particularly on motion captioning—the task that most directly reflects whether the learned representations capture movement semantics. This suggests that random factorization orders do not sufficiently exploit the natural ordering structure of everyday behavior as self-supervision, and that autoregressive prediction along the natural temporal order is important for learning meaningful movement representations.

D.2 Effect of global translation on motion forecasting and interpolation

We hypothesize that the relatively large errors of the baseline methods in motion forecasting and interpolation are partly attributable to their learned representations preserving global translation information less effectively than those learned by BAL.

To test this hypothesis, we conduct an oracle global-translation alignment experiment. For each forecasted or interpolated sequence, we compute the pelvis offset between the first predicted frame

Table 7: Effect of oracle global-translation alignment on motion forecasting and interpolation. BAL remains the best in most settings after alignment.

Method	**Forecasting** (mm) ↓			**Interpolation** (mm) ↓		
	AMD	Nym	E3D	AMD	Nym	E3D
A4Mer	633.9	285.1	375.5	163.7	120.8	86.3
PUMPS	163.2	130.3	102.2	182.6	154.3	87.0
USDRL	133.3	115.9	87.9	117.7	106.7	59.2
BehaveMAE	170.2	169.1	126.1	135.6	103.4	77.8
MotionBERT	282.4	244.1	162.0	135.2	129.8	69.2
MacDiff	152.7	77.9	71.5	106.9	100.4	55.4
BAL	87.8	77.7	63.7	117.5	77.0	59.2

and its ground-truth counterpart and translate the entire predicted sequence by this offset. For each method and evaluation setting, we report the lower of the original and aligned errors, ensuring that the alignment does not penalize methods for which it is unhelpful.

Tab. 7 reports the results of this oracle alignment experiment. Compared with the original errors in Tab. 1, the alignment substantially reduces the performance gap between BAL and the baselines, providing evidence that global translation errors account for part of this gap.

Importantly, however, this alignment does not eliminate BAL's advantage. Even under this oracle alignment, which particularly benefits methods with large global translation errors, BAL still outperforms the baselines in most evaluation settings, and the pairwise ordering between BAL and each baseline remains unchanged from Tab. 1. Thus, although global translation errors contribute to the baselines' larger errors, they do not fully explain BAL's advantage.

Note that because this alignment relies on ground-truth poses, which are unavailable at inference time, the original results in Tab. 1 remain the appropriate measure of practical forecasting and interpolation performance.

D.3 Effect of chunk length

In BAL, we set the chunk length to 4 seconds based on the definition of atomic actions in Nymeria [25]. The chunk length controls the temporal scale represented by each latent token and is the only hyperparameter in BAL involved in the design of the supervisory signal. Unlike augmentation strengths or masking scales in prior methods, however, it has a direct semantic interpretation: it specifies the temporal unit over which movement is abstracted, and can therefore be chosen intuitively based on the expected duration of meaningful actions.

To examine how this temporal granularity affects the learned representations, we pretrain BAL with different chunk lengths and evaluate the resulting representations on all downstream tasks. As shown in Tab. 8, performance is consistently strong across the tested range, with every variant either outperforming existing self-supervised methods or remaining competitive with them. Thus, BAL achieves strong downstream performance without requiring careful tuning of the chunk length for each task.

The results also reveal a modest dependence on the task. Shorter chunks generally perform better for motion forecasting and interpolation, whereas longer chunks tend to benefit motion captioning. This tendency is consistent with the nature of the downstream tasks. Because forecasting and interpolation are evaluated directly in pose space, they benefit from fine-grained pose information, which shorter chunks are more likely to retain. In contrast, captioning requires higher-level semantic understanding of movements, for which longer temporal units can provide a more suitable abstraction. When further gains are desired, the chunk length can therefore be chosen according to an intuitive notion of the temporal granularity relevant to the target task.

Table 8: Effect of the chunk length on motion captioning, forecasting, and interpolation. "Best prior result" denotes, for each metric, the best result among prior self-supervised methods. Across the tested chunk lengths, BAL outperforms or remains competitive with the best prior result. We use a 4-second chunk length in the main text.

Method	**Chunk length**	**Captioning** ↑								**Forecasting** (mm) ↓			**Interpolation** (mm) ↓		
		BLEU-1		ROUGE-L		CIDEr		BERTScore							
		Nym	E3D	Nym	E3D	Nym	E3D	Nym	E3D	AMD	Nym	E3D	AMD	Nym	E3D
BAL	2 sec	37.4	58.7	41.0	45.0	62.6	16.9	28.0	34.2	89.1	72.2	61.4	110.6	70.3	55.0
BAL	3 sec	33.4	58.9	40.2	45.1	63.6	17.7	28.7	34.6	95.1	82.2	63.4	111.3	75.2	56.3
BAL	4 sec	37.3	58.9	41.2	45.3	63.1	18.6	29.6	35.0	90.5	82.1	63.8	117.5	77.0	59.2
BAL	5 sec	41.4	59.1	41.7	45.2	71.8	18.3	31.6	34.5	115.3	87.5	66.4	118.3	80.7	59.6
Best prior result	-	36.7	58.8	41.4	45.3	65.6	18.4	28.5	35.2	136.1	359.0	73.2	112.7	103.5	56.6

Table 9: Comparison of parameter counts and inference latency for the representation extractors.

Method	**#Params [M]**	**Latency [ms]**
A4Mer	9.64	16.0
PUMPS	53.01	8.1
USDRL	95.15	9.2
BehaveMAE	5.81	4.3
MotionBERT	42.42	21.4
MacDiff	11.96	5.3
BAL	14.18	12.2

Table 10: Parameter-matched comparison with scaled-up baselines. Methods that originally have fewer parameters than BAL are scaled to approximately match BAL's parameter count. BAL still outperforms them on most metrics.

Method	**#Params [M]**	**Captioning** ↑								**Forecasting** (mm) ↓			**Interpolation** (mm) ↓		
		BLEU-1		ROUGE-L		CIDEr		BERTScore							
		Nym	E3D	Nym	E3D	Nym	E3D	Nym	E3D	AMD	Nym	E3D	AMD	Nym	E3D
A4Mer	14.66	6.4	54.2	7.7	40.7	2.1	12.4	3.3	27.1	235.2	313.4	136.0	135.8	109.4	73.9
BehaveMAE	14.19	30.9	59.1	37.8	44.6	47.9	17.8	26.0	34.1	171.0	139.0	116.4	139.1	98.3	76.9
MacDiff	14.33	39.0	59.1	41.3	45.0	62.4	18.5	28.7	34.1	142.9	367.1	84.5	118.3	404.1	58.1
BAL	14.18	37.3	58.9	41.2	45.3	63.1	18.6	29.6	35.0	90.5	82.1	63.8	117.5	77.0	59.2

D.4 Trainable parameters and inference latency

Tab. 9 reports the parameter counts and inference latency of the representation extractors used at inference for BAL and all compared methods. Inference latency is measured on a single NVIDIA A100 GPU with a batch size of 1, and may vary substantially depending on implementation details.

For a fairer comparison with methods that have fewer parameters than BAL, we additionally scale up these methods by increasing the number of layers to approximately match BAL's parameter count and evaluate the resulting models on each downstream task. As shown in Tab. 10, BAL still outperforms these methods on most metrics, showing that its gains are not simply due to model scale.

E Additional qualitative results

We show additional qualitative results of motion retrieval (Figs. 6 and 7), captioning (Fig. 8), forecasting (Fig. 9), and interpolation (Fig. 10).

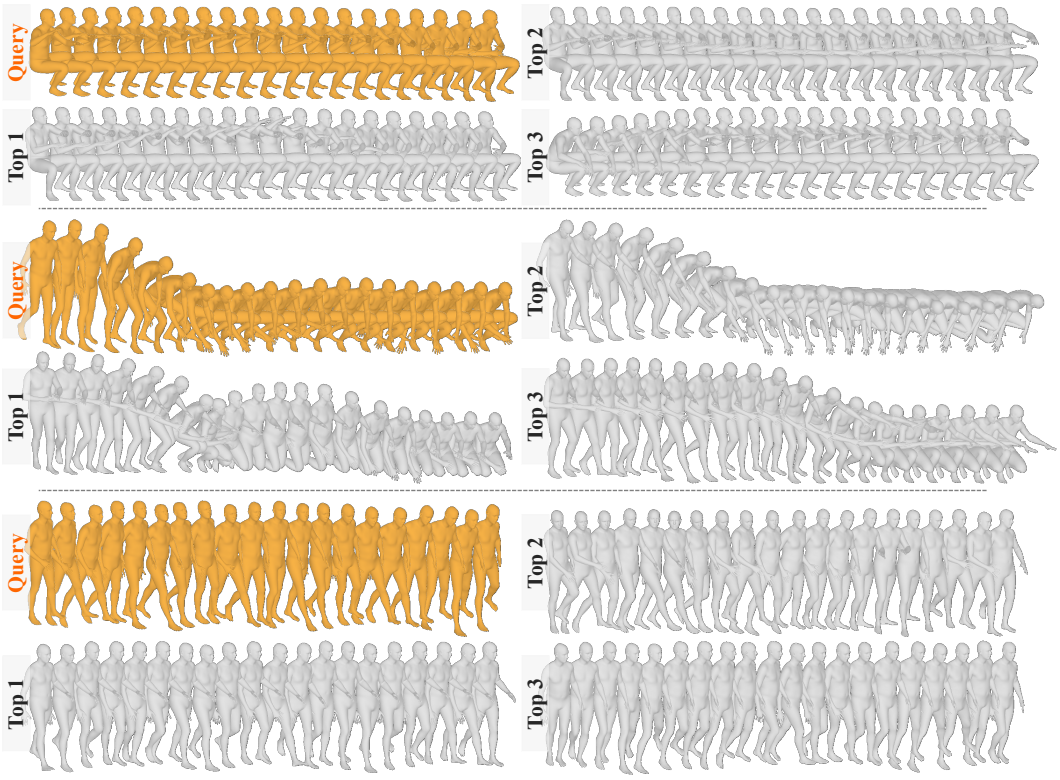


Figure 6: Motion retrieval examples. The nearest neighbors in the representation space correspond to semantically similar movements.

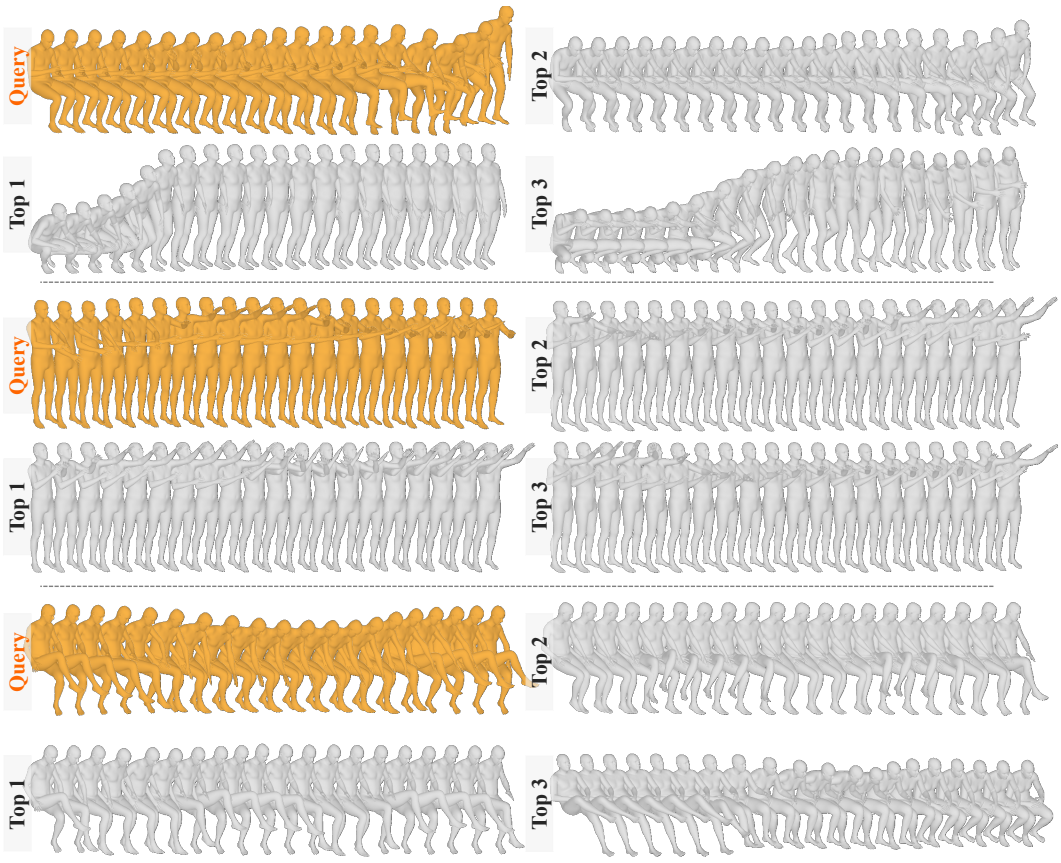


Figure 7: Additional motion retrieval examples.



GT: The person is in the kitchen, alternately shaking her feet while talking to her peers then turns around, following her peer's instruction.

BAL: The person is following the instructions of her peer in the living area.

MacDiff: The person is dancing in the bedroom.

MotionBERT: The person is in the living room and kitchen.

USDRL: The person continues to stir the food in the pan.



GT: The person is preparing the meal in the kitchen.

BAL: The person is cooking in the kitchen.

MacDiff: The person is playing foosball with his peer in the activity area.

BehaveMAE: The person is playing foosball with her peer in the living area.

A4Mer: is standing in the living room of the room in the kitchen area the kitchen area of the living room of the room where the man is standing in the living room of the room where the man stands in the living room of the room where the man stands in the living room of the room where the man stands in the



GT: The person is tidying up the living area.

BAL: The person continues to arrange the items on the center table in the living room.

MacDiff: The person is searching for something in the living area.

PUMPS: The person is playing a game with his peers in the living area.

A4Mer: then she is sitting in the living room as she is playing jenga with her friends in the living room.



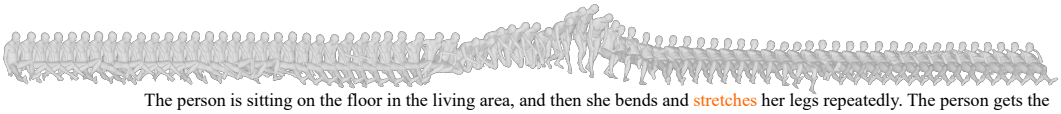
GT: The person is cooking meat in the kitchen while chatting with her peer.

BAL: The person is talking to her peer while cooking in the kitchen.

MacDiff: The person is writing on the whiteboard in the meeting room.

PUMPS: The person is playing a game with his peers in the living area.

USDRL: The person is standing in the kitchen area talking to her peer.



GT: The person is sitting on the floor in the living area, and then she bends and stretches her legs repeatedly. The person gets the napkin on the floor and puts it on the table, and then she stands up, lies down on the sofa while following the instructions of her peer. The person then touches her glasses.

BAL: The person continues to perform body stretch in the bedroom.

MacDiff: The person is sitting on the floor in the living area while watching television.

BehaveMAE: The person is sitting on the floor in the living area while his peers are watching him.

MotionBERT: The person is sitting on the sofa in the living area while watching TV.

Figure 8: Qualitative results of motion captioning with the frozen LLM.

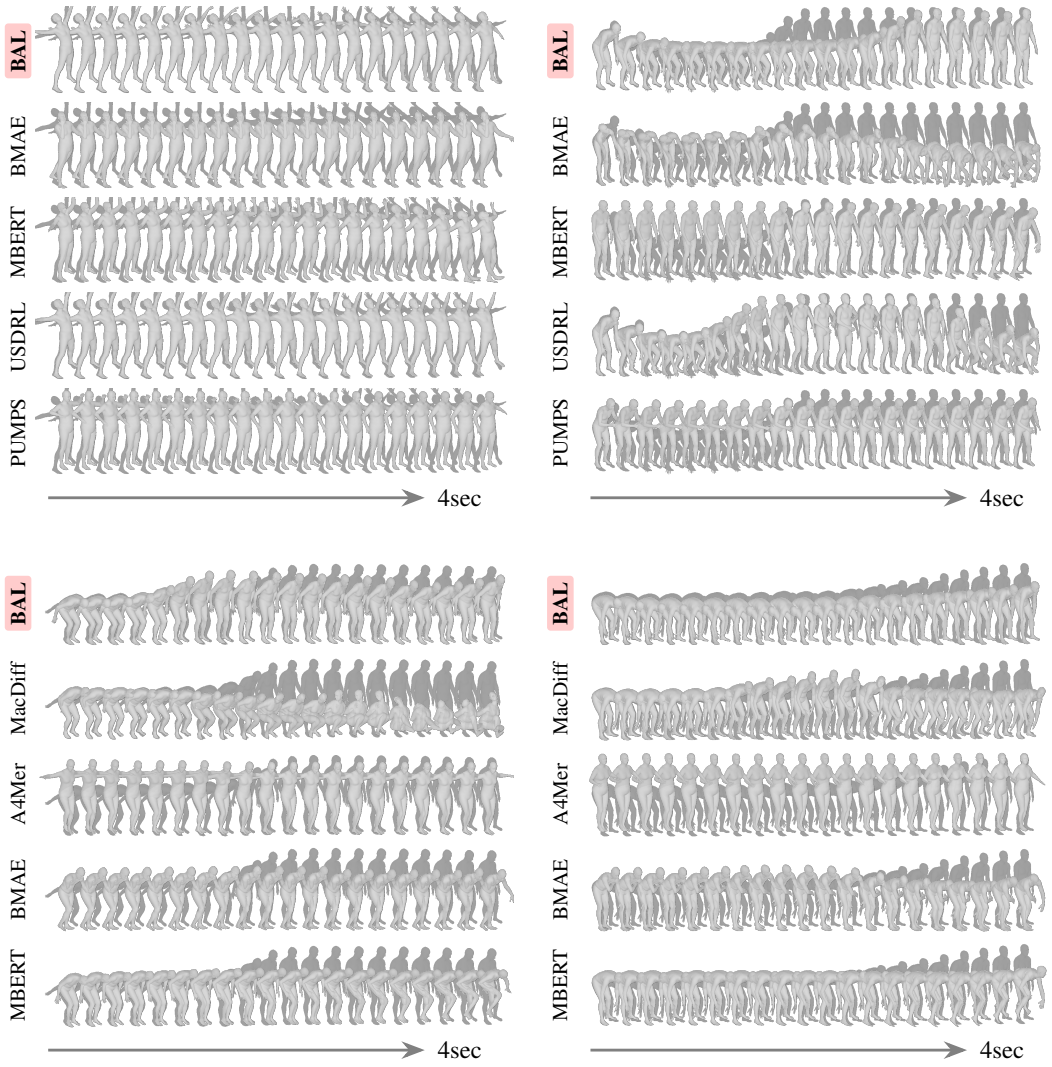


Figure 9: Forecasted poses via autoregressive latent token prediction and decoding. The SMPL meshes in dark color denote the ground truth.

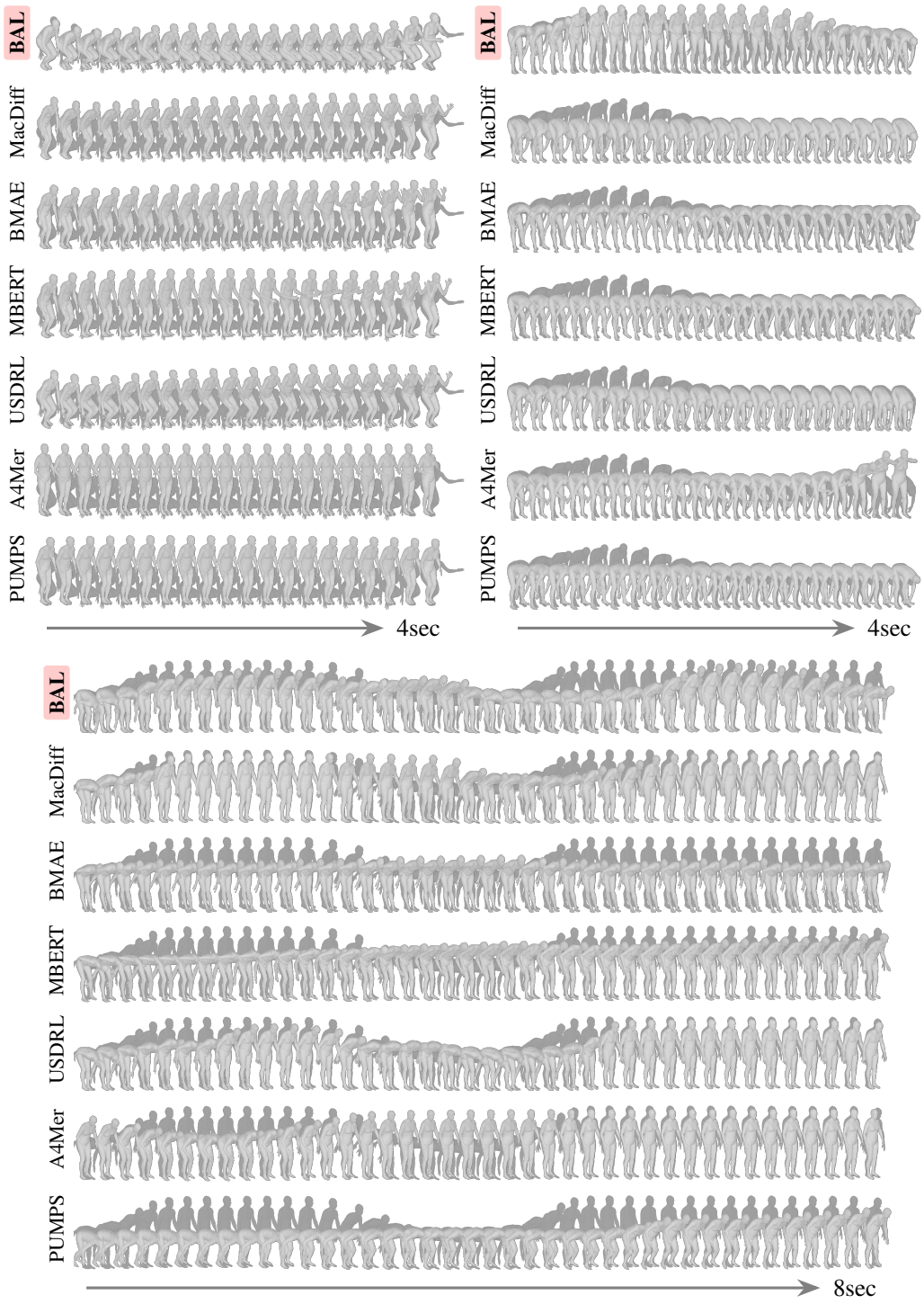


Figure 10: Interpolated poses via latent token interpolation and decoding. The dark-colored SMPL meshes are the ground truth.

F Broader impact

BAL provides a foundational framework for learning semantic representations of human movement from 3D pose sequences, contributing to advances in human behavior modeling. These representations may support applications involving semantic understanding of human movement, such as human-computer interaction, assistive technologies, rehabilitation analysis, and robotics. At the same time, improved movement representations could be misused for unauthorized surveillance or privacy-invasive behavioral profiling, especially if applied to data collected without consent or adequate safeguards. Although this work focuses on 3D pose sequences, similar principles may also be explored in other sequential modalities. Applications and extensions of this work should be developed with careful attention to consent, data privacy, fairness, and domain-specific ethical guidelines.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main claims and scope of our learning framework, which are supported by the theoretical analysis in Sec. 3.3 and the experimental results in Sec. 4, including Tabs. 1 and 2.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the need to manually choose the chunk length as a limitation and analyzes its effect in Sec. D.3.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The main theoretical claim is stated and derived in Secs. 3.3 and B.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The proposed method is described in Sec. 3, with pseudocode provided in Alg. 1. Sec. 4 describes the datasets and evaluation protocols, while Tabs. 3 and 5 provide the architecture and training settings.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We plan to release our code and pretrained weights.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Dataset splits and preprocessing are described in Sec. 4, and optimizer, learning rates, batch sizes, schedules, and training iterations are reported in Tab. 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Due to computational constraints, the main comparisons in Tab. 1 and most ablations in Tab. 2 were conducted with a single run. For the autoregressive training and inference variants in Tab. 2, we report the mean and standard deviation over 10 runs of downstream head training with different random seeds, keeping the pretrained representation extractors fixed.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute resources and training time for pretraining and each downstream task are reported in Tab. 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This work uses existing datasets cited in Sec. 4 and does not introduce procedures that deviate from the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Sec. F discusses both positive applications of BAL for human movement understanding and potential negative risks such as surveillance and privacy-invasive behavioral profiling.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a high-risk generative model, scraped dataset, or other asset that requires special safeguards beyond standard responsible release practices.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We checked the licenses of the models and datasets used in our experiments and cite these assets appropriately.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper introduces BAL and documents its architecture, training settings, and pseudocode in the appendix. We also plan to release the source code.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve new crowdsourcing experiments or newly collected human-subject data; it uses existing datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve newly conducted human-subject research by the authors; it relies on existing datasets.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The paper uses an LLM for the motion captioning evaluation, as described in Sec. 4.2, but the LLM is not an important, original, or non-standard component of the core method.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.