MoChat: Joints-Grouped Spatio-Temporal Grounding Multimodal Large Language Model for Multi-Turn Motion Comprehension and Description

Jiawei Mo, Yixuan Chen, Rifen Lin, Yongkang Ni, Feng Liang, Min Zeng, Xiping Hu, Min Li

***Abstract*— Despite continuous advancements in deep learning for understanding human motion, existing models often struggle to accurately identify action timing and specific body parts, typically supporting only single-round interaction. This limitation is particularly pronounced in home exercise monitoring, neurological disorder assessment, and rehabilitation, where precise motion analysis is crucial for ensuring exercise efficacy, detecting early signs of neurological conditions, and guiding personalized recovery programs. In this paper, we propose MoChat, a multimodal large language model capable of spatio-temporal grounding of human motion and multi-turn dialogue understanding. To achieve this, we first group spatial features in skeleton frames according to human anatomical structures and process them through a Joints-Grouped Skeleton Encoder. The encoder's outputs are fused with large language model embeddings to generate spatio-aware representations. A cross-attention-based Regression Head module is then designed to align hidden-layer embeddings and skeletal sequence embeddings, enabling precise temporal grounding. Furthermore, we develop a pipeline for temporal grounding task to extract timestamps from skeleton-text pairs and construct a multi-turn instruction dialogues for spatial grounding task. Finally, various task instructions are generated for jointly training. Experimental results demonstrate that MoChat achieves state-of-the-art performance across multiple metrics in motion understanding tasks, making it as the first model capable of fine-grained spatio-temporal grounding of human motion.**

***Index Terms*— Large language model, motion analysis, multimodal, skeleton, spatiotemporal phenomena**

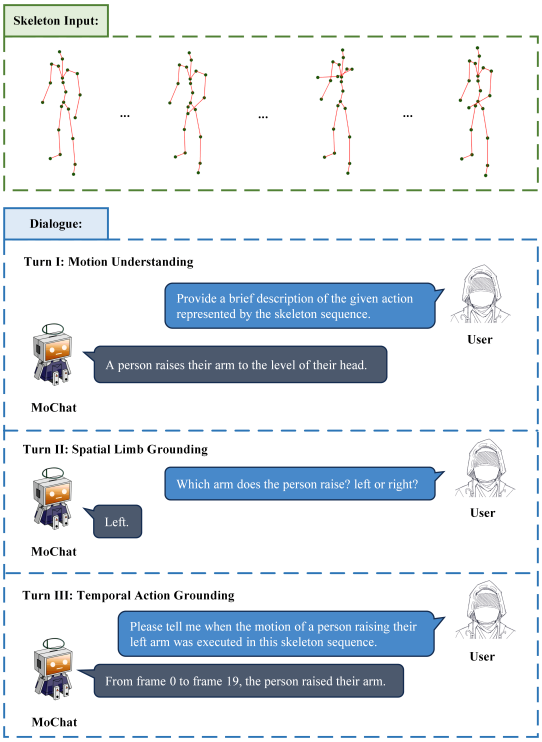


Fig. 1. Illustration of the multi-turn spatio-temporal grounding capabilities of MoChat. MoChat is a MLLM designed for motion comprehension, with capabilities that extend beyond regular motion description. Specifically, MoChat can follow user instructions to summarize motion sequences (Turn I), pinpoint specific body parts involved in the motion (Turn II), and ground the start and end frames corresponding to user queries (Turn III).

I. Introduction

This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFA0706202. *(Corresponding author: Xiping Hu; Min Li.)*

Jiawei Mo, Yixuan Chen, Rifen Lin, Min Zeng, Min Li are with the School of Computer Science and Engineering, Central South University, Changsha 410083, PR, China (e-mail: mojiawei@csu.edu.cn; csucyx@csu.edu.cn; rifen_lin@csu.edu.cn; zengmin@csu.edu.cn; limin@mail.csu.edu.cn).

Yongkang Ni is with the School of Software, Xinjiang University, Urumqi 830046, PR China (e-mail: 107552301686@stu.xju.edu.cn).

Feng Liang, Xiping Hu is with the School of Medical Technology, Beijing Institute of Technology, Beijing 100081, China, and also with the Department of Engineering, Shenzhen MSU-BIT University, Shenzhen 518172, China (e-mail: fliang@smbu.edu.cn; huxp@bit.edu.cn).

THE intricate analysis and comprehension of human motion hold significant promise across various domains, particularly in the realm of healthcare. Applications range from enhancing patient rehabilitation protocols, refining surgical techniques, monitoring neurological disorders, to advancing sports medicine and telehealth services. Several commercial platforms have incorporated skeleton-based motion tracking into clinical practice. Kaia Health and SWORD Health provide musculoskeletal rehabilitation services using smartphone-based or wearable-assisted pose tracking, while Reflexion

Health's VERA system enables camera-guided home-based recovery after joint replacement.

Recent research has shown growing interest in skeleton-based models for clinical and rehabilitation applications. Marusić et al. [1] used a Transformer model to classify specific exercise errors in low back pain rehabilitation, enabling targeted feedback in remote healthcare or telerehabilitation contexts. Martinel et al. [2] proposed SkelMamba, which decomposes motion into spatial and temporal streams to enhance sensitivity to motor abnormalities, further validating skeleton-based analysis for neurological assessment. With the advent of Multimodal Large Language Models (MLLMs) such as Flamingo [3], GPT-4V [4], and CogVLM [5], there has been a significant shift in the ability of AI to interpret multimodal inputs—including human motion—within an open-vocabulary framework. Existing works on MLLM-based human motion understanding can be broadly classified into two categories: the first category encompasses models focused on RGB image and video understanding, such as VideoChat [6] and BLIP-2 [7], which are not specifically tailored for human motion understanding tasks; the second category comprises specialized models designed explicitly to interpret human motion from motion capture data, showcasing advanced performance in analyzing motion, exemplified by TM2T [8] and MotionGPT [9]. However, these models still struggle to accurately ground specific time periods and body parts involved in motion, which limits their performance in motion understanding tasks.

The challenge of building such motion understanding models lies in accurately modeling the relationships between motion sequences and captions, and incorporating the temporal dimensions essential for understanding motion. For the first challenge, recent research [10] has demonstrated the efficacy of pre-trained Large Language Models (LLMs) in modeling relationships between diverse non-textual modalities and textual data. Specifically, motion sequences can be regarded as a unique form of language. By utilizing an projector, these sequences can be fine-tuned to facilitate the conversion of motion information into descriptive text. Additionally, in the action recognition field, studies [11], [12] have shown that grouping keypoints can enhance the representation of action features. For the second challenge, existing video captioning models [13], [14] are capable of extracting the time intervals in videos that correspond to specific captions. Therefore, it is promising to train a model capable of locating the spatial and temporal positions of specific action sequences.

In this work, we propose MoChat, a MLLM that is capable of spatio-temporal grounding in human motion understanding, facilitated by multi-turn dialogue context. To enable the model's understanding of motion sequences, we first pre-train a Transformer-based [15] skeleton encoder. The keypoints are partitioned into four groups based on the human anatomical structure for motion encoding, enhancing the encoder's geometric perception. The resulting motion features are then converted through a lightweight projector into LLM-compatible tokens, which are subsequently combined with text instruction tokens as input into the LLM. This allows the model to comprehend the semantics of the motion sequence and generate descriptive text for the motion sequence. Meanwhile, by calculating the similarity between the LLM's hidden states and the motion tokens, the temporal boundaries corresponding to the text are regressed. Additionally, to construct dialogue data for training, we develop a pipeline for extracting timestamps from the motion caption datasets, and create multi-turn spatial dialogues by keyword matching. Using the resulting multi-task instruction set, we conduct a two-stage joint training of MoChat, which enhances its detailed action understanding capabilities in both temporal and spatial dimensions. We validate our model through extensive experiments on the HumanML3D dataset [16], covering the tasks of Motion Understanding, Spatial Limb Grounding, and Temporal Action Grounding, evaluated using tranditional metrics and GPT-4. The results demonstrate that MoChat achieves state-of-the-art performance, highlighting its fine-grained spatio-temporal motion understanding capabilities. Our contributions can be summarized as follows:

1) We propose MoChat, a motion understanding MLLM that comprehends motion sequences, accurately captions the movement of specific body parts, and precisely identifies the time boundaries corresponding to user instructions. To the best of our knowledge, MoChat is the first MLLM capable of spatio-temporal grounding of actions in skeleton sequences.
2) We develop a semi-automated pipeline to extract timestamps from the motion caption datasets, and construct multi-turn spatial dialogues, both of which are used to create a multi-task instruction set for joint training.
3) Comprehensive experiments validate the advanced motion understanding capabilities of MoChat, demonstrating its spatial and temporal grounding abilities. Our model introduces functionalities not found in existing motion understanding models, making it more versatile and user-friendly.

II. Related Work

A. Motion Understanding Models

Motion understanding tasks can generally be categorized into fixed-class action recognition, which involves a predefined set of classes, and open-vocabulary motion understanding, which does not restrict the number of classes. In the branch of fixed-class action recognition, numerous skeleton-based methods have been proposed. [17], [18], [19] For instance, ST-GCN [20] applies 3D graph convolution to human skeleton sequences across both temporal and spatial dimensions to extract action features. With the rise of self-supervised learning and Transformers [15], there has been a shift towards exploring Transformer-based self-supervised action recognition. [21], [22] One such method is GL-Transformer [23], which constructs pretext tasks for amplitude and displacement recovery using the relative and absolute positions of joints, enabling effective representation of skeleton sequences without reliance on action labels.

With the advancement of LLMs, open-vocabulary motion understanding tasks have become feasible. The models typically involve a motion encoder combined with a language model to comprehend motion sequences. A notable example

is TM2T [8], which employs VQVAE [24] to obtain discrete motion tokens from a codebook. These motion tokens and their corresponding text tokens are then fed into simple neural machine translators (NMT) for both motion-to-text and text-to-motion conversion, enabling bidirectional matching. MotionGPT [9] and AvatarGPT [25] replace NMT with LLMs equipped with projector, fine-tuned with instructions to enable understanding and generation of motion sequences under various conditions. However, these approaches have not fully harnessed the comprehension capabilities of LLMs, primarily due to inadequate training instructions and the constrained representational power of the encoders. In contrast, our proposed MoChat employs more than three instruction sets related to temporal and spatial tasks for joint training, and utilizes a Transformer-based skeletal encoder to extract motion features, thereby demonstrating superior motion understanding capabilities.

B. Vision-Language Models

The development of LLMs has significantly advanced the field of vision-language models, with notable progress in both image-language models [4], [26] and video-language models [13], [27].

In the domain of image-language models, BLIP-2 [7] pretrains a BERT-based [28] Q-Former to align visual and textual information, using a fixed-length learnable query vector to extract semantic information from images. However, this approach overly compresses the information, limiting the model's ability to capture intricate image details. LLaVA-1.5 [26] employs VIT [29] as the image encoder and Vicuna [30] as the language decoder. A lightweight projector is used to map image embeddings into the language latent space, enabling LLMs to understand visual content. In contrast, CogVLM [31] introduces a visual expert module that is equivalent in size to the LLM. Yet this approach doubles the inference parameters of the MLLM, which presents challenges during deployment.

For video understanding, ChatUnivi follows the LLaVA's projector approach, also compressing information by aggregating dynamic visual tokens across different frames. On the other hand, TimeChat adopts the InstructBLIP [32] strategy to encode temporal information through textual instructions. Besides, it employs a sliding window to segment video frames, encoding them with multiple Q-Formers. These approaches enhance TimeChat's temporal awareness but it struggles with continuous temporal concept comprehension.

Additionally, previous work [33] has revealed significant challenges in vision models' handling of "geometry-aware" semantic correspondences. For example, these models often misinterpret spatial relationships, such as confusing the left and right sides of the image with the left and right sides of the objects within it, which hampers their spatial grounding capabilities. This spatial orientation ability is crucial in the field of sports medicine, especially in assessments such as the asymmetry evaluation in Parkinson's disease, where the model needs to make accurate judgments about the left and right limbs.

Addressing the aforementioned limitations, the MoChat model we propose is capable of pinpointing the continuous start and end times of actions in the temporal dimension, while also demonstrating commendable discriminative ability between left and right limbs in the spatial dimension. This renders the MoChat model particularly advantageous in healthcare applications.

III. MoChat: A Chat MLLM for Motion

In this section, we introduce MoChat, a MLLM capable of spatio-temporal grounding in human motion understanding, facilitated by multi-turn dialogue context. The inclusion of two novel modules, the Joints-Grouped Skeleton Encoder (JGSE) and the Regression Head (RH), enhances MoChat's ability to finely understand motions and accurately ground the start and end frames of instruction-corresponding motions. To further empower MoChat to follow human instructions and understand context in complex multi-turn, multi-task dialogues, we construct such dialogues for spatial fine-grained motion understanding and develop a pipeline for timestamp extraction. Based on these dialogues, we perform a two-stage integrated instruction tuning on a pre-trained LLM to create MoChat.

A. Overall Framework

As illustrated in Fig. 2, MoChat is composed of a spatio-aware JGSE, a LLM equipped with projector, and a RH. Given an input skeleton sequence with T frames, $X_s = \{X_s^t\}_{t=1}^{T}$, the skeleton encoder JGSE first extracts motion features while maintaining the same sequence length. Then, a projector converts these features into motion tokens H_s, which are mapped to the language latent space. These motion tokens H_s are concatenated with input instruction tokens H_t and fed into a LLM. The LLM's final hidden states H_m are then decoded into appropriate responses, which are passed to a regression head to obtain the corresponding timestamps simultaneously.

1) Joints-Grouped Skeleton Encoder: Previous transformer based models typically apply positional encoding to skeleton joints based on the specific order determined by the joint numbering scheme. However, different skeleton types have different joint numbering orders, which forces models to undergo retraining when the skeleton type changes. Whereas this approach is effective for handling specific skeleton types, it ultimately limits the model's ability to generalize and effectively represent other skeleton types. In transformers, position embeddings are initially designed to reinforce the positional relationships within a sequence, making the order of the input sequence critically important. This implies that when a frame of skeleton joints is used as the input sequence, different orders of the joints can significantly alter the transformer's encoding output.

Taking this consideration into account, we propose the JGSE module, which builds upon [23] by incorporating a novel position encoding strategy. For each skeleton frame t, which contains M joints denoted as $X_s^t = \{j_k\}_{k=1}^{M}$, we partition the joints j_k into four anatomical groups $g \in \{\text{Arm (A)}, \text{Leg (L)}, \text{Trunk (Tr)}, \text{Global Joint (GJ)}\}$. The Global Joint (GJ) is computed as a weighted combination

This article has been accepted for publication in IEEE Journal of Biomedical and Health Informatics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JBHI.2025.3631045

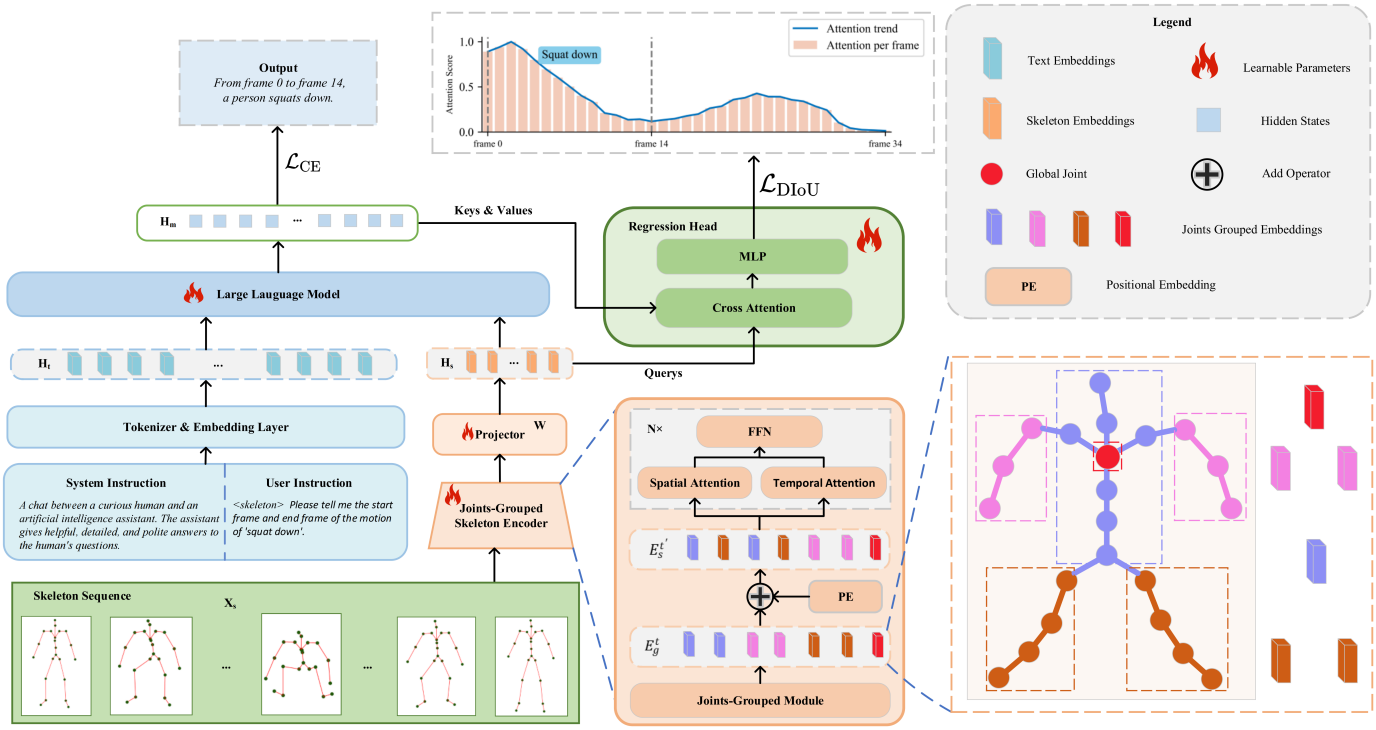


Fig. 2. Overview of MoChat. Given a skeleton motion sequence as input, **(a) Joints-Grouped Skeleton Encoder** first extracts motion features by grouping and embedding the joints separately. Then, **(b) Projector** converts these features into motion tokens $\boldsymbol{H_s}$ in the language latent space. These motion tokens $\boldsymbol{H_s}$ are concatenated with instruction tokens $\boldsymbol{H_t}$ and input to a **(c) Large Language Model (LLM)**. The LLM's final hidden states $\boldsymbol{H_m}$ are decoded into appropriate responses and passed to a **(d) Regression Head** to obtain the corresponding timestamps.

of all joints to capture a holistic view of the entire pose. Each group is independently embedded to produce four group-level embeddings: E_A^t, E_L^t, E_{Tr}^t, and E_{GJ}^t. These are then concatenated to form the full embedding of frame t:

$$E_g^t = \text{Concat}(E_A^t, E_L^t, E_{Tr}^t, E_{GJ}^t). \tag{1}$$

We denote the full sequence of per-frame embeddings as $E_g = \{E_g^t\}_{t=1}^T$, where T is the number of frames. Next, spatial and temporal position embeddings are added to E_g to produce E_s, enriching the embeddings with structural and sequential context. To facilitate the exchange of information aggregated to the joints, E_s is restored to the original joint ordering via an index-based reordering operation, yielding E'_s. The sequence E'_s is then passed to an N-layer Transformer encoder that performs spatio-temporal attention across both joints and frames.

This enables a two-stage encoding process, where localized dynamics are first captured within anatomical groups, followed by global spatiotemporal interaction over the restored joint sequence. Unlike prior works that apply attention directly over flat or grouped joint sequences without restoring their spatial layout [23], [34], our approach preserves both anatomical structure and positional semantics across stages, forming a principled architecture for skeleton representation.

2) Language Module: Inspired by recent advances in vision-language models such as LLaVA [26] and Chat-UniVi [27], which insert visual tokens into the input stream of LLMs, we adopt a similar token-level fusion strategy. However, MoChat applies this paradigm to a different modality—3D skeleton-based motion sequences—introducing unique challenges in temporal encoding and multimodal alignment. Specifically, we employ a Vicuna-based LLM equipped with a trainable linear projector. The motion features E''_s extracted from the JGSE are mapped into the language embedding space via a projection matrix W, resulting in a motion token sequence $H_s \in \mathbb{R}^{T\times d}$ that preserves temporal structure.

As illustrated in Fig. 2, we prepend a fixed system instruction to guide the LLM toward motion understanding tasks. The user input is denoted as a variable instruction containing a placeholder `<skeleton>` to indicate where motion tokens should be inserted. The full instruction is tokenized and embedded to obtain H_t, and the motion embeddings H_s are inserted at the placeholder position, yielding a fused sequence that is passed to the LLM.

The final hidden states H_m from the LLM are projected by a linear layer to produce token logits $\mathbf{z}$, which are decoded into MoChat's response X_o. During training, we apply a cross-entropy loss between $\mathbf{z}$ and the shifted ground truth response X_{gt}^{id}.

$$\mathcal{L}_{\text{CE}} = -\sum_i X_{gt}^{\text{id}(i)} \log \sigma(\mathbf{z}^{(i)}), \tag{2}$$

where $\sigma(\cdot)$ denotes the softmax function. Notably, the inserted skeleton tokens do not contribute to this loss, as they are treated as context.

3) Regression Head: For precisely grounding the time boundaries, we design a regression head, which is responsible

for predicting the start frame $\mathrm{ID}_{\text{start}}$ and the end frame ID_{end}. To compute the start and end frame IDs corresponding to the language, we naturally consider calculating the similarity between the motion embedding tokens H_s and the LLM hidden states H_m. In this process, the motion embedding tokens H_s are fed into the regression head as $Queries$, while the LLM hidden states H_m serve as $Keys$ and $Values$. We employ the scaled dot-product attention mechanism to compute the attention weights:

$$\mathrm{W}_{\text{cross}} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right), \tag{3}$$

where Q represents the queries, K represents the keys, and d_k is the dimension of the keys. We deliberately let the motion embeddings H_s act as $Queries$, because the cross-attention output then lives in the same temporal space that the regression head is required to predict. Keeping this alignment allows the MLP to map the attended feature into the start and end frame indices in a single step, without any extra pooling or coordinate transformation.

In the resulting attention weight matrix $\mathrm{W}_{\text{cross}} \in \mathbb{R}^{T\times N}$, we focus specifically on the weights associated with the `[BOS]` token, denoted as $W_0 \in \mathbb{R}^{T\times 1}$. The `[BOS]` token is an indicative marker that signifies the beginning of a sequence and typically carries significant contextual information about the entire sequence. Consequently, we consider it to be of paramount importance for representing the sequence as a whole, and we utilize a Multi-Layer Perceptron (MLP) to regress the start and end frame IDs:

$$\text{IDs} = \text{MLP}(W_0^T \cdot H_s), \tag{4}$$

where W_0^T is the transpose of weight vector W_0. $H_s \in \mathbb{R}^{T\times D}$ represent the motion embedding tokens, where D is the hidden dimension of the LLM. The output IDs is a two-element vector corresponding to the start and end frame indices, represented as $[\mathrm{ID}_{\text{start}}, \mathrm{ID}_{\text{end}}]$.

Then, for stable convergence, the DIoU loss [35] between the predicted and ground truth IDs is calculated as:

$$\mathcal{L}_{\text{DIoU}} = 1 - \left(\text{IoU} - \frac{d^2(\mathrm{ID}_{\text{start}}, \mathrm{ID}_{\text{end}}, \mathrm{ID}^{\text{gt}}_{\text{start}}, \mathrm{ID}^{\text{gt}}_{\text{end}})}{c^2(\mathrm{ID}_{\text{start}}, \mathrm{ID}_{\text{end}}, \mathrm{ID}^{\text{gt}}_{\text{start}}, \mathrm{ID}^{\text{gt}}_{\text{end}})}\right), \tag{5}$$

where IoU denotes the Intersection over Union. The $d^2(\cdot)$ term represents the squared Euclidean distance between the center points of the predicted and ground truth intervals, while the $c^2(\cdot)$ term normalizes this distance by the square of the length of the union interval. Subsequently, the final loss is a combination of the cross-entropy loss and the DIoU loss:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda_{\text{DIoU}}\mathcal{L}_{\text{DIoU}}, \tag{6}$$

where λ_{DIoU} is a hyperparameter that balances the two losses.

B. Data Construction

We construct motion understanding dialogues using the motion caption dataset. Table I presents all the templates used to construct the dialogues. We firstly design instruction templates such as "*Provide a brief description of the given action represented by the skeleton sequence*" and directly use the corresponding motion caption as the answer for constructing basic motion understanding dialogues. To further enhance the model's perception of movement, we also design dialogue templates from the temporal and spatial dimensions.

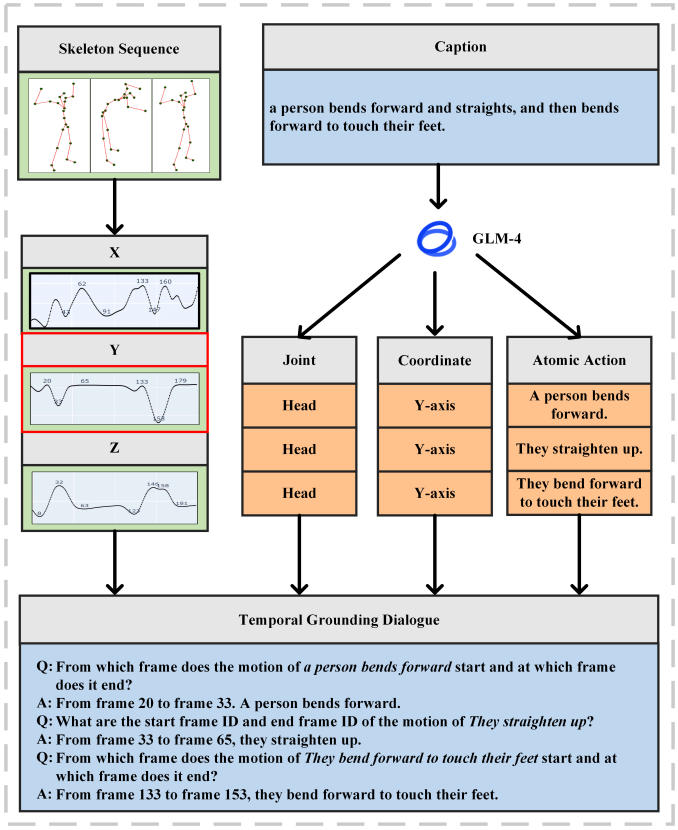


Fig. 3. Pipeline for constructing Temporal Grounding Dialogues. GLM-4 splits the caption into atomic actions and identifies the corresponding most significant joint and coordinate. The curves represent the coordinates of the selected joint, with the numbers on the curves indicating the frame IDs of the extremum points. We construct multi-turn temporal grounding dialogues based on the final extracted results.

1) Timestamps Extraction Pipeline: As illustrated in Fig. 3, we develop a pipeline for extracting timestamps from skeleton sequences based on textual annotations. To avoid any potential bias in subsequent GPT-4 scoring, GLM-4 [36] is employed to determine the atomic action referenced in the captions and to identify one corresponding joint and axis (X for left-right, Y for height, Z for front-back) exhibiting the most significant variation. This process simplifies the task of accurately assigning timestamps to each individual action. The selection of joints and axes is further refined based on motion data. Following the analysis from GLM-4, the selected motion data is first smooth-filtered. Subsequently, extreme points and the differences between them are computed, allowing for the identification of the start and end frame IDs that correspond to the atomic action with the maximum variation. After extraction, a manual review is conducted, and the results are used to construct the temporal grounding dialogues as shown in Table I.

2) Spatial Dialogues Construction: The process for constructing the Spatial Dialogue is illustrated in Fig. 4. We

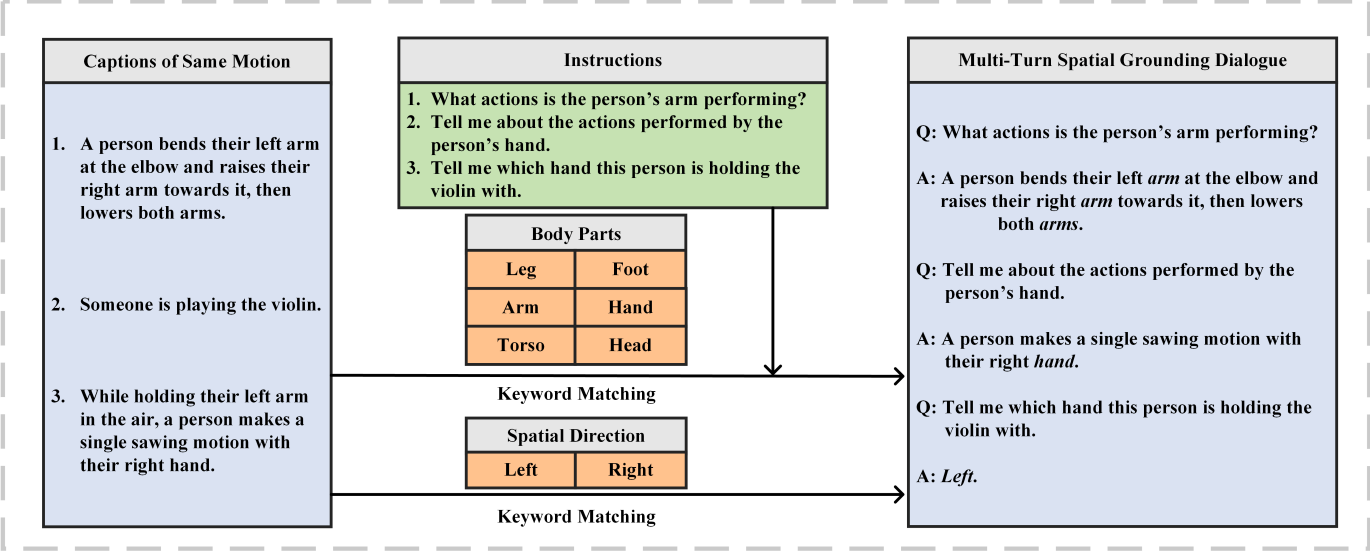


Fig. 4. The process of constructing Spatial Dialogues. We construct multi-turn spatial grounding dialogues by matching clauses within captions of the same motion based on selected keywords, and then combining them with pre-set instruction templates.

construct multi-turn dialogues for spatial fine-grained motion using keyword matching. First, we select keywords such as *foot*, *leg*, *hand*, *arm* and *torso* based on human anatomical structure. Next, we create instruction templates, as shown in Table I, where the `<body_part>` placeholder in the instruction can be replaced with these keywords. Captions containing the corresponding keywords are then selected as responses. For spatial relationships, we design gap-filling dialogues based on captions that include spatial keywords such as *left* and *right*. Specifically, we ensure a balanced distribution of different answers to prevent model bias. If a caption involves multiple body parts, it is divided into separate dialogue turns, with each turn focusing on describing the motion of a single body part, thereby constructing a multi-turn dialogue that captures the entire movement.

C. Training Strategy

Our training strategy consists of three stages: pre-training the skeleton encoder, aligning motion-language embeddings, and end-to-end fine-tuning of the model. The latter two stages involve an integrated instruction tuning process on a pre-trained LLM, with the JGSE module kept frozen.

To provide a clearer illustration of the three-stage training strategy, we include a schematic diagram in Fig. 5 and a corresponding pseudo-code summary in Algorithm 1. In Stage 0, we pre-train the JGSE skeleton encoder in an unsupervised manner on skeleton sequences, following the data preprocessing and pretext tasks outlined in [23]. Here, σ_{gt} and δ_{gt} denote the pseudo-labels for motion magnitude and 3D direction constructed from the skeleton sequences, and L_σ, L_δ represent the corresponding cross-entropy losses.

In Stage 1, we jointly train the projector and regression head while keeping the LLM frozen. The goal is to align motion embeddings with language embeddings using a multi-task instruction set. We merge the instruction-response dialogues constructed in the previous subsection and randomly sample a batch at each iteration. Human instructions and motion sequences are treated as loss-irrelevant inputs to the LLM, while the dialogue responses serve as loss-relevant targets. Autoregressive training is applied to predict the next token in the responses. Timestamps are extracted from the ground truth responses and used to compute the DIoU loss.

In Stage 2, we fine-tune the entire model, including the LLM, projector and RH modules, using the same instruction data to further enhance task performance.

IV. Experiments

A. Dataset and Evaluation Metrics

1) Dataset: The HumanML3D dataset [16], which contains 14,616 motion sequences and 44,970 motion captions, is used for training and evaluation. This dataset is divided into training, validation, and test sets, with 80%, 5%, and 15% of the data allocated to each set, respectively. We construct the multi-task dialogues from the caption sets and use them in conjunction with the corresponding 22-joint SMPL [37] skeleton sequences as input for model training and evaluation.

The KIT Motion–Language (KIT–ML) dataset [38] comprises 3,911 motion clips (approximately 11.2 hours) paired with 6,278 free-form English sentences. Each clip contains 21-joint skeleton sequences sampled at 10 Hz. Following the same protocol used for HumanML3D, we split the dataset into training, validation, and test sets with an 80%, 5%, and 15% ratio, respectively. To ensure consistency and comparability, we use the 3D joint position sequences released by TM2T, and apply the same filtering strategy described in their code, which excludes clips shorter than 24 frames or longer than 200 frames.

2) Evaluation Metrics: We evaluate our model on three tasks: Motion Understanding, Spatial Limb Grounding, and Temporal Action Grounding. For the Motion Understanding

TABLE I
DIALOGUE TEMPLATES.

Basic Motion Understanding Dialogues	
Instruction Templates	Provide a brief description of the given action represented by the skeleton sequence. Write a terse but informative summary of the action depicted by the skeleton sequence. Share a concise interpretation of the action demonstrated in the skeleton sequence. Relay a brief, clear account of the action shown in the skeleton sequence. Render a clear and concise summary of the action sequence. Create a compact narrative representing the action portrayed in the skeleton sequence. Give a short and clear explanation of the subsequent action depicted by the skeleton sequence. Summarize the movement content of the action demonstrated by the skeleton sequence. Describe the action concisely as represented in the skeleton sequence. Offer a succinct explanation of the action presented in the skeleton sequence. Present a compact description of the action sequence's key features.
Example	**Q:** Provide a brief description of the given action represented by the skeleton sequence. **A:** A person walks forward, then turns around and walks backward.
Temporal Grounding Dialogues	
Dialogue Templates	From which frame does `<motion>` start and at which frame does it end? What are the start frame ID and end frame ID of `<motion>`? Please tell me when `<motion>` was executed in this skeleton sequence. From `<frameid_1>` to `<frameid_2>`, `<motion>`.
Example	**Q:** Please tell me when ***A person bends forward*** was executed in this skeleton sequence. **A:** From ***frame 20*** to ***frame 33***. ***A person bends forward***.
Spatial Gap-filling Dialogues	
Instruction Templates	`<motion_with_gap>`, Complete the content in brackets with ***left*** **or** ***right***.
Example	**Q:** Person leans forward goes onto knees whilst first putting (____) hand on ground for support and stays on knees. Complete the content in brackets with left or right. **A:** ***Left***.
Spatial Multi-turn Dialogues	
Instruction Templates	Describe the movements of the person's `<body_part>` in detail. Please provide details about the actions of the person's `<body_part>`. What actions is the person's `<body_part>` performing? Tell me about the actions performed by the person's `<body_part>`.
Example	**Q:** Tell me about the actions performed by the person's ***torso***. **A:** The person walked backwards slowly, their ***torso*** remaining upright, before stepping forward with a forceful kick. **Q:** What actions is the person's ***arm*** performing? **A:** A person bends their left arm at the elbow and raises their right ***arm*** towards it, then lowers both arms.

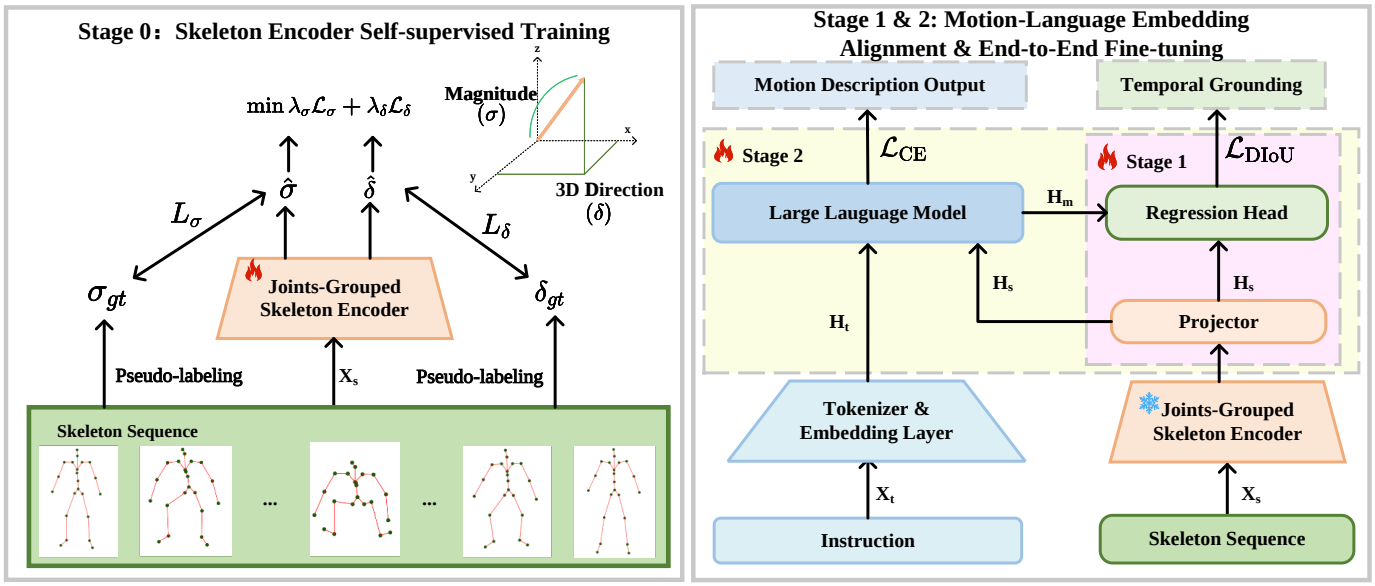


Fig. 5. Three-Stage Training Strategy for MoChat

Algorithm 1 Three-Stage Training Strategy

Input: Skeleton sequences X_s, instruction dialogues $X_t = \{(\text{instr}, \text{resp})\}$
Output: Trained MoChat model

Stage 0: Skeleton Encoder Pre-training
for each batch X_s **do**
  Generate pseudo-labels σ_{gt}, δ_{gt} from X_s
  $\hat{\sigma}, \hat{\delta} = \text{JGSE}(X_s)$
  Compute loss: $\mathcal{L} = \lambda_\sigma \cdot L_\sigma + \lambda_\delta \cdot L_\delta$
  $\theta_{\text{JGSE}} \leftarrow \text{AdamW}(\nabla_{\theta_{\text{JGSE}}} \mathcal{L})$
end for
Freeze JGSE

Stage 1 & 2: Instruction-Tuning with Projector and Regression Head
for each batch (X_s, X_t) **do**
  $H_s = \text{Projector}(\text{JGSE}(X_s))$
  $H_t = \text{Embedding}(\text{Tokenizer}(X_t))$
  $H_m = \text{LLM}(\text{Concat}(H_t, H_s))$
  $X_o = \text{Decoder}(H_m)$
  $\text{IDs} = \text{RH}(H_m, H_s)$
  Compute loss: $\mathcal{L} = L_{\text{CE}} + L_{\text{DIoU}}$
  $\theta_{\text{proj}}, \theta_{\text{RH}} \leftarrow \text{AdamW}(\nabla_\theta \mathcal{L})$
  if Stage 2 **then**
    $\theta_{\text{LLM}} \leftarrow \text{AdamW}(\nabla_{\theta_{\text{LLM}}} \mathcal{L})$
  end if
end for

task, we follow the approach in [8], utilizing standard linguistic metrics including BLEU [39], ROUGE [40], CIDEr [41], and BERTScore [42]. In addition, we incorporate GPT4Score, a recently proposed automatic evaluation metric that uses GPT-4 as an expert judge to assess the quality of generated responses [43]. GPT4Score belongs to the family of "learned" or LLM-based evaluators, which aim to capture semantic alignment and contextual consistency beyond surface-level lexical overlap. This complements traditional metrics by better reflecting human judgment in open-ended language generation tasks. For the Spatial Limb Grounding task, we use accuracy as the evaluation metric, as the spatial test set is based on gap-filling dialogues. For the Temporal Action Grounding task, the evaluation metric is "R@1, IoU = μ," which denotes the percentage of retrieved frame IDs with an intersection over union (IoU) greater than μ compared to the ground truth.

B. Implementation Details

All our models employ the AdamW optimizer for training. For the skeleton encoder pre-training, we use a batch size of 128 and train the model for 120 epochs with a learning rate of 5×10^{-5} and a decay rate of 0.99. The encoder consists of a 4-layer transformer. The input sequences are padded to 500 frames with a value of 99.9. We adopt the pre-trained Vicuna-v1.5-13B model [30] as the language foundation model. In the stage of aligning the motion-language embeddings, we train the projector and regression head with a batch size of 64 for 3 epochs, using a learning rate of 2×10^{-3}. The learning rate schedule includes a warm-up ratio of 0.03, followed by cosine annealing. The weight parameter of loss λ_{DIoU} is set to 5. In the final stage, for fine-tuning the model end-to-end, a batch size of 128 is applied, with training conducted over 1 epoch at a learning rate of 2×10^{-5}. The same warm-up and cosine annealing schedule from the previous stage is utilized. All models are trained on 8 $\times$ Nvidia A800 GPUs. When GPU memory is insufficient, we reduce the per_device_train_batch_size and increase the gradient_accumulation_steps while keeping the product of per_device_train_batch_size, GPU_num, and gradient_accumulation_steps equal to the original batch size.

C. Comparisons with State-Of-The-Art Methods

We evaluate MoChat with state-of-the-art methods on three task including Motion Understanding, Spatial Limb Grounding and Temporal Action Grounding. The model that employs the original GL-Transformer skeleton Encoder (GLTE) [23] is referred to as the baseline, with the LLM component consistent across all models. The model that includes both the JGSE and RH modules is referred to as MoChat-R, while the model without the RH module is referred to as MoChat.

1) Comparisons on Motion Understanding: The Motion Understanding task involves generating a brief caption based on a given motion sequence. We directly adopt the linguistic results from [25] and utilize the suggested evaluation method to assess MoChat. For a fair comparison, when assessing the GPT4Score metric, we utilize the motion data as specified in [9] to regenerate captions for evaluation. Models that are not open-sourced cannot generate captions, and therefore, no evaluation results are available for them. As shown in Table II, the $\uparrow$ symbol indicates that a higher value is better, with bold and underline used to denote the best and second-best results, respectively. We evaluate the performance of MoChat and other models on the HumanML3D and KIT-ML datasets. On the HumanML3D dataset, MoChat significantly outperforms recent works on the Motion Understanding task across most metrics, except for the BERTScore. AvatarGPT achieves a higher BERTScore, possibly due to its training on a large corpus of text generated by LLMs like ChatGPT. This enhances its capability to produce captions with a richer vocabulary and more nuanced language structure. Although the semantic alignment with the ground truth is high, the textual fidelity, as measured by metrics like BLEU or CIDEr, may be significantly lower. On the KIT-ML dataset, we evaluated TM2T using its official testing script and released weights, and evaluated MoChat using the same evaluation implementation to ensure a fair comparison. As shown in Table II, MoChat outperforms TM2T across all evaluation metrics, demonstrating superior adaptability to different motion domains.

2) Comparisons on Spatial Limb Grounding: The Spatial Limb Grounding task involves identifying which body part is responsible for the action in a given motion sequence. Following the data processing methods outlined in previous sections, we constructed 2,574 gap-filling questions from the HumanML3D test set to evaluate the model. Since current skeleton-based motion understanding models lack spatial

TABLE II
COMPARISON OF THE MOTION UNDERSTANDING TASK ACROSS DATASETS.

Datasets	Methods	BLEU@1 ↑	BLEU@4 ↑	ROUGE ↑	CIDEr ↑	BERTScore ↑	GPT4Score ↑
HumanML3D	TM2T [8]	48.90	7.00	38.10	16.80	32.20	–
	MotionGPT [9]	48.20	12.47	37.40	29.20	32.40	5.14
	AvatarGPT [25]	49.28	12.70	40.44	32.65	**53.58**	–
	Baseline	59.81	19.26	45.86	45.09	43.57	5.21
	MoChat (Ours)	**61.75**	**21.60**	**47.59**	**51.57**	45.59	**5.99**
	MoChat-R (Ours)	60.06	21.30	46.08	46.57	42.56	5.25
KIT-ML	TM2T [8]	35.35	5.70	33.94	12.07	38.62	4.58
	MoChat (Ours)	**36.89**	**5.89**	**35.34**	**14.13**	**41.18**	**5.23**

grounding capabilities and instruction-following ability, we evaluated GPT-4V, a vision-language model, for comparison. To ensure fair comparison, we rendered the SMPL-based 3D skeleton sequences into RGB human motion videos, and uniformly sampled 10 frames to serve as GPT-4V's input. This approach aligns with the input expectations of GPT-4V and avoids introducing unfair preprocessing gaps. Skeleton-based models such as MotionGPT were not included in this task because they cannot follow spatial instructions nor produce outputs suitable for metric-based evaluation. As shown in Table III, MoChat achieves the highest accuracy of 85.70%, demonstrating its strong capability in spatial limb grounding.

TABLE III
COMPARISON OF SPATIAL LIMB GROUNDING TASK ON SPATIAL TEST SET.

Model	Acc. ↑
GPT-4V [4]	68.02
Baseline	80.12
MoChat (Ours)	**85.70**
MoChat-R (Ours)	81.90

3) Comparisons on Temporal Action Grounding: The Temporal Action Grounding task requires the model to accurately locate the time range corresponding to a queried action. We constructed a test set containing 233 samples to evaluate models' performance. Since existing skeleton-based models do not possess the temporal instruction-following capability needed for this task, we rendered the motion sequences into video format and employed TimeChat, a time-sensitive video-language model, as a comparative baseline. Although TimeChat does not exhibit strong general-purpose instruction-following ability, it is specifically designed to predict temporal boundaries, making it suitable for this evaluation. In the base MoChat, the model is trained to generate start and end frame indices in natural language form, relying solely on the LLM's generative ability. In contrast, MoChat-R includes a dedicated regression head that numerically predicts the action boundaries, yielding higher precision. As shown in Table IV, although MoChat-R slightly underperformed MoChat in the previous two tasks, it outperformed other models in the Temporal Action Grounding task.

TABLE IV
COMPARISONS OF TEMPORAL ACTION GROUNDING TASK ON TEMPORAL TEST SET.

Model	R@1 (IoU=0.5) ↑	R@1 (IoU=0.7) ↑
TimeChat [13]	2.10	0.40
Baseline	12.45	6.87
MoChat (Ours)	19.31	5.58
MoChat-R (Ours)	**21.89**	**12.02**

D. Ablation Studies

We conduct ablation studies on different configurations, including alternative modules, training datasets, and training methodologies, to verify the robustness and effectiveness of our method across various settings. The results are presented in Table V, Table VI, and Table VII. The BMUD, SD, and TGD indicate the instruction training sets of Basic Motion Understanding Dialogue, Spatial Dialogue and Temporal Grounding Dialogue, respectively. The notation BMUD+SD+TGD or BST signifies that the model was trained jointly on these combined instruction sets. In the table, 'Stage 1' refers to the stage of aligning the motion-language embeddings, while 'Stage 2' denotes the stage of full fine-tuning of all parameters within the LLM. The presentation of results follows the same conventions outlined earlier in the text.

1) Different Datasets Configurations: During the fine-tuning of the LLM, we observe catastrophic forgetting, where the model lost its ability to follow general instructions, a capability typically possessed by the original chat model. Specifically, To preserve the model's instruction-following ability, we utilize the Puffin dataset, a subset of processed ShareGPT data, containing 3,000 examples, with each response generated using GPT-4. As shown in Table V, when both the GLTE module and the BMUD training set are utilized, not employing the Puffin dataset leads to superior results in the metrics for the Motion Understanding task. However, the model fails to generate reasonable responses to other types of instructions, such as "Who are you?"—a question unrelated to the Motion Understanding task—resulting in a less user-friendly model.

We also perform ablation experiments utilizing incrementally combined instruction training sets across the Motion Understanding task and the Spatial Limb Grounding task mentioned above. In the Motion Understanding task, for most models, the best performance is achieved by combining BMUD, SD, and TGD in Stage 1 and Stage 2 of training, while for models equipped with the GLTE module, the best results

TABLE V
ABLATION STUDY ON THE MOTION UNDERSTANDING TASK ACROSS DIFFERENT MODULES AND TRAINING CONFIGS.

Modules	LoRA	Stage 1	Stage 2	BLEU@1 ↑	BLEU@4 ↑	ROUGE ↑	CIDEr ↑	BERTScore ↑	GPT4Score ↑
GLTE	–	BMUD wo Puffin		**62.36**	**22.51**	47.09	50.35	44.25	5.53
GLTE	r=64 alpha=16	BMUD		37.42	7.54	32.01	21.81	38.46	5.24
GLTE	–	BMUD		59.85	20.80	45.46	44.88	41.63	4.74
JGSE	–			61.36	21.30	46.69	47.98	44.14	5.62
JGSE+RH	–			60.11	20.34	45.86	46.45	42.84	5.10
GLTE	–	BMUD+SD		59.95	20.51	**47.64**	49.30	44.28	5.40
JGSE	–			60.81	20.87	47.04	50.60	44.60	5.96
JGSE+RH	–			60.31	20.64	45.87	46.65	42.84	5.19
JGSE+CFT	–	BMUD	BST	59.96	20.88	46.38	47.11	43.47	5.50
GLTE	–	BMUD+SD+TGD		59.81	19.26	45.86	45.09	43.57	5.21
JGSE	–			61.75	21.60	47.59	**51.57**	**45.59**	5.99
JGSE+CFT	–			61.16	21.49	46.75	49.27	44.12	**6.05**
JGSE+RH	–			60.06	21.30	46.08	46.57	42.56	5.25

are obtained by combining only BMUD and SD. In Table VI, for the Spatial Limb Grounding task, the performance of all models is optimal when trained in conjunction with the integration of the three training datasets. These suggest that, in most cases, training a model with a combination of multiple datasets constructed for different tasks can better harness the model's capabilities, leading to enhanced performance on the same task.

2) Different Modules Configurations: In addition to the RH module, we also experiment with using Custom Frame ID Tokens (CFT) to identify the start and end frames corresponding to the captions. Specifically, we introduce T new tokens into the tokenizer's vocabulary, such as `<frameid_0>`, `<frameid_1>`, ..., `<frameid_T-1>`. These tokens are then associated with their corresponding embeddings. We add these frame ID embeddings to the motion token embeddings, similar to the role of position embeddings, before inserting them into the language embeddings.

As indicated in Table V, Table VI, and Table VII, under the condition of an equivalent training set, the model employing the JGSE module yields the best performance for both the Motion Understanding and Spatial Limb Grounding tasks, suggesting that the JGSE module is well-suited for these types of tasks. In the temporal action grounding task, the model that combines the JGSE and RH modules achieves the highest performance, while the model using the JGSE and CFT modules ranks as the second-best. This demonstrates the functional efficacy of the RH module within the context of the temporal action grounding task. To further evaluate our design choices, we examined an alternative attention configuration inspired by HiLM-D [44], in which the roles of H_s and H_m are swapped—i.e., the LLM hidden states serve as $Queries$, and the motion embeddings act as key-value inputs (denoted as RHR). As shown in Table VII, this reversal leads to a noticeable performance drop: R@1 decreases from 21.89 to 13.70 at IoU = 0.5, and from 12.02 to 4.70 at IoU = 0.7. We attribute this degradation to two factors: (1) the enlarged query length imposed by H_m, which introduces computational overhead and optimization difficulty, and (2) the need for an additional pooling operation to collapse token-level outputs back to the timeline, which weakens the precision of frame-wise grounding. These results empirically confirm that anchoring the query in the motion-temporal domain is essential for accurate and efficient grounding.

3) Different Training Configurations: We investigate approaches to decrease resource consumption by experimenting with LoRA. As depicted in Table V, we train and evaluate the model using a LoRA rank of 64 and an alpha value of 16. However, compared to the fully fine-tuned model, although memory usage is reduced, the resultant performance degradation is unacceptable. This suggests the necessity to explore alternative, more efficient strategies for diminishing memory usage.

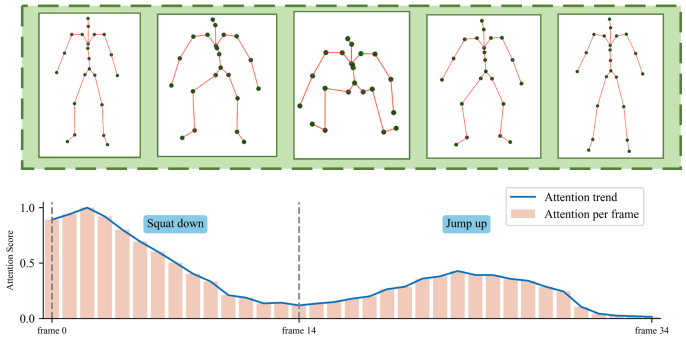
Additionally, we explore the impact of using different instruction training sets at different stages of instruction fine-tuning. For instance, to reduce training time, we apply the BMUD training set during the alignment of motion-language embeddings and combine the BMUD, SD, and TGD training sets during the full fine-tuning stage. As shown by the results in Table V, Table VI, and Table VII, when the same training set is employed across both stages, the model equipped with the JGSE and CFT modules performs better on the Motion Understanding and Spatial Limb Grounding tasks, but worse on the Temporal Action Grounding task.

TABLE VI
ADDITIONAL EXPERIMENTS FOR SPATIAL LIMB GROUNDING TASK.

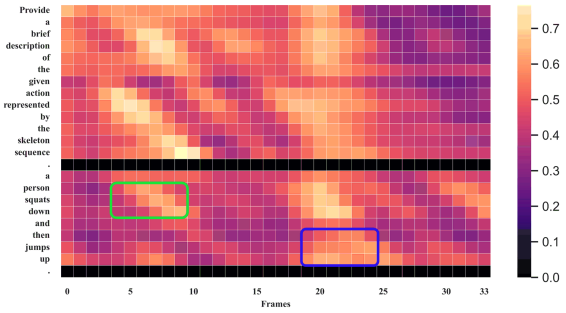
Modules	Stage 1	Stage 2	Acc. ↑
GLTE	BMUD+SD		77.66
JGSE			81.05
JGSE+RH			79.02
JGSE+CFT	BMUD	BST	85.28
GLTE	BMUD+SD+TGD		80.12
JGSE			**85.79**
JGSE+CFT			85.70
JGSE+RH			81.90

V. ANALYSIS

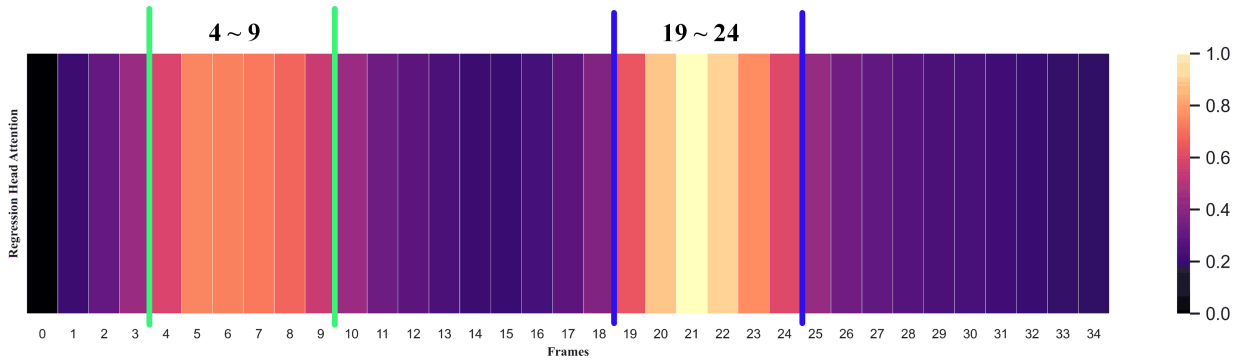
User: Provide a brief description of the given action represented by the skeleton sequence.
MoChat: A person squats down, then jumps up.
User: Please tell me the start frame and end frame of the motion of a person squats down.
MoChat: From frame 0 to frame 14, the person squats down.
User: From which frame does the motion of a person jumps up start and at which frame does it end?
MoChat: From frame 19 to frame 24, the person jumps up.



(a) Temporal attention weights of Joints-Grouped Skeleton Encoder (JGSE).



(b) Attention map of the LLM.



(c) Cross-attention weights of the Regression Head (RH).

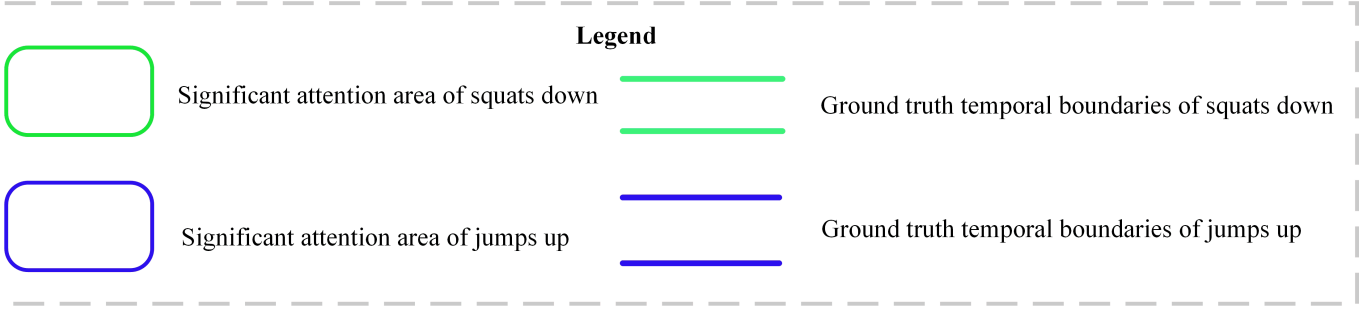


Fig. 6. Attention visualization of three modules. The x-axis represents the frame ID in the time dimension of motion sequences. The y-axis in (a) and the color in (b) and (c) both indicate the magnitude of attention weights. A brighter color signifies higher attention or activation. The y-axis in (b) represents words and punctuation marks in the dialogue.

TABLE VII
ADDITIONAL EXPERIMENTS FOR THE TEMPORAL ACTION GROUNDING TASK.

Modules	Stage 1	Stage 2	R@1(IoU=0.5) ↑	R@1(IoU=0.7) ↑
JGSE+CFT	BMUD	BST	20.17	9.01
JGSE	BMUD+SD+TGD		19.31	5.58
JGSE+CFT			18.03	7.30
JGSE+RHR			13.70	4.70
JGSE+RH			**21.89**	**12.02**

A. Analysis of Learned Attention

To gain further insights into our model, we visualize the attention weights of the JGSE, LLM, and RH modules. For the JGSE, we compute the average self-attention weights from the last layer of the Transformer Encoder and then visualize the attention of the last temporal `[CLS]` token to other skeleton frame tokens, as shown in Fig. 6 (a). We concatenate the resulting motion embeddings with the language embeddings and feed them into the LLM, then extract the attention matrix from the first head of the first layer. The attention weights are averaged across multiple language tokens to form complete words, as depicted in Fig. 6 (b). For the Regression Head, we visualize the cross-attention weights of the `[BOS]` token with respect to the motion embeddings, as shown in Fig. 6 (c). The examples in Fig. 6 illustrate that our model exhibits higher self-attention and cross-attention weights at the corresponding frame and word positions for the actions "squat down" and "jump up." This confirms the model's effective capture of temporal awareness and motion-caption mapping, enabling it to successfully perform the Temporal Action Grounding task.

B. Analysis of Hallucination

When a LLM is fine-tuned on a small-scale, domain-specific dataset, it is prone to generating inaccurate or fabricated responses, commonly referred to as hallucinations. Therefore, evaluating the model's robustness to hallucinations is essential for ensuring the reliability of its outputs. Therefore, we demonstrate the robustness of MoChat from both qualitative and quantitative perspectives.

1) Qualitative Analysis: First, from a qualitative perspective, Table VIII presents three representative examples of motion-captioning results to qualitatively compare our model (MoChat-R) with MotionGPT. Each input motion sequence is shown along with its ground truth caption (top row), the caption generated by MotionGPT (middle row), and the caption generated by our model MoChat-R (bottom row). In the first example, the ground truth describes a person interacting with a handrail using their right hand. While MoChat-R successfully captures this fine-grained action ("holding handrail with right hand"), MotionGPT hallucinates an unrelated scene ("walking downhill") that is not supported by the input motion, indicating a semantic drift from the visual evidence. In the second example, the hallucination is subtler: MotionGPT mentions a "grey block," which is not present in the ground truth description. This reference arises from a coincidental artifact in the 3D visualization (i.e., the rendered floor), which is not part of the actual motion semantics. MoChat-R, by contrast, accurately focuses on the relevant motion content ("jumps forward with both arms outstretched") without being misled by visualization artifacts. In the third example, both models correctly describe the circular walking motion, but only MoChat-R explicitly captures the "clockwise" direction, aligning more precisely with the reference caption. These examples highlight MoChat-R's robustness against hallucinations, especially when compared to MotionGPT. By aligning motion understanding more closely with actual observed sequences and avoiding overfitting to visual noise or unintended cues, MoChat-R demonstrates stronger semantic grounding and improved caption fidelity.

2) Quantitative Analysis: From a quantitative perspective, GPT4Score has been suggested as an indicator of a model's tendency to hallucinate in generated outputs [45] . The final column of Table II reports the GPT4Score for various models on the Motion Understanding task across two datasets. On the HumanML3D dataset, both MoChat and MoChat-R achieve higher GPT4Scores than Baseline and MotionGPT, indicating better alignment with human-like judgments and reduced hallucination. Similarly, on the KIT-ML dataset, MoChat outperforms TM2T in terms of GPT4Score, further confirming its robustness. These results quantitatively support the conclusion that MoChat exhibits enhanced robustness against hallucinations, aligning well with reference captions and avoiding semantically inaccurate or fabricated content.

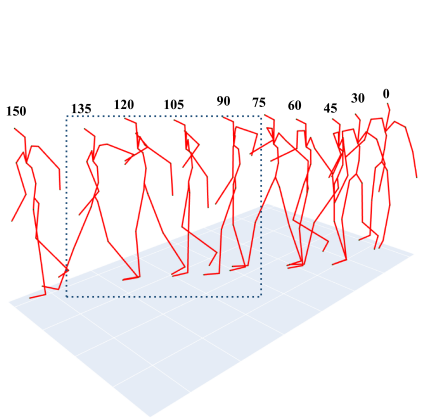
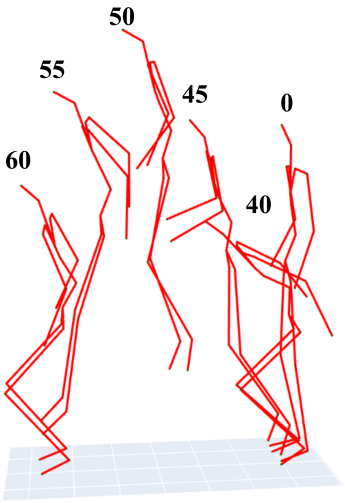
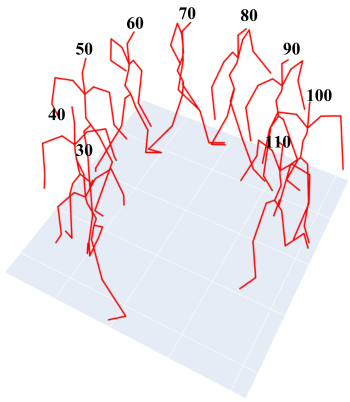
VI. DISCUSSION

1) Overall Performance Comparison: Across all three evaluation tasks, MoChat establishes a new performance bar among public skeleton-based methods. On the Motion Understanding task, MoChat achieves the best overall performance across all reported metrics, including BLEU-4, CIDEr, and GPT4Score. Compared to published skeleton-based methods such as TM2T, MotionGPT, and AvatarGPT, MoChat provides substantial gains—e.g., +9.1 BLEU-4 and +19.0 CIDEr over MotionGPT on HumanML3D. On KIT-ML, MoChat also achieves higher scores than TM2T across all metrics, with a modest gain of 0.19 BLEU-4, indicating consistent generalization. Although AvatarGPT achieves a higher BERTScore, we attribute this to its large-scale LLM-generated text corpus. In contrast, MoChat emphasizes faithful captioning, as evidenced by higher n-gram and sentence-level metrics, indicating reduced hallucination.

2) Spatial Reasoning Capabilities: In the Spatial Limb Grounding task, MoChat achieves 85.7% accuracy, outperforming GPT-4V by 17.68%. We argue that explicit skeletal encoding, rather than purely RGB-based vision models, enables more fine-grained spatial limb grounding—an ability not yet explored by existing vision–language systems or motion-language models like MotionGPT.

3) Temporal Grounding Performance: MoChat's RH module plays a key role in Temporal Action Grounding. MoChat-R substantially outperforms TimeChat with over 5x higher R@1 scores under standard IoU thresholds (0.5 and 0.7), demonstrating its superior temporal grounding ability. An

TABLE VIII
THE QUALITY RESULTS OF MOCHAT-R AND THE STATE-OF-THE-ART METHOD ON THE MOTION UNDERSTANDING TASK. THE RESULTS DEMONSTRATE THAT OUR METHOD EXHIBITS A STRONGER PERCEPTION OF ACTION DETAILS. ITALICS IN THE TABLE INDICATE THE MATCHED DETAILS.

Input Motion Sequences			
Caption	a person takes a step forward, moves to theor right, them continues foward with their ***right hand on a rail.***	a person jumps forward once.	a person walks in a circle, ***clockwise.***
MotionGPT	a person is walking downhill.	a person jumps down a grey block.	a person walks in a circle.
MoChat-R	a person walks forward while ***holding handrail with right hand.***	a person jumps forward with both arms outstretched.	a person walks in a ***clockwise*** circle.

ablation study shows that removing RH results in a 5% drop, highlighting the importance of regression head—a design choice absent in existing baselines.

4) Modularity and Skeleton-Centric Design: These results demonstrate that a modular skeleton-to-language framework can rival or outperform complex video-centric architectures. Unlike holistic vision models, MoChat operates on structured joint representations, allowing for interpretable, anatomy-aware motion analysis.

5) Training Insights: Our ablation also reveals that mixing generic instruction tuning data (e.g., Puffin) with task-specific motion-language corpora helps mitigate catastrophic forgetting during Stage 2 fine-tuning. This insight may inform future work on domain-adaptive instruction tuning for motion-based LLMs.

VII. CONCLUSION

In this paper, we present MoChat, a multimodal large language model that comprehends motion sequences, accurately captions the movement of specific body parts, and precisely identifies the time boundaries corresponding to user instructions. To the best of our knowledge, MoChat is the first MLLM capable of spatio-temporal grounding of actions in single skeleton sequences.

Although MoChat has its limitations, particularly in terms of real-time performance and resource consumption when compared to fixed-class action recognition models, it has carved out a new path in the field of medical applications. Specifically, in domains such as home exercise monitoring, neurological disorder assessment, and rehabilitation therapy, MoChat's ability to provide precise motion analysis is revolutionary. It is essential for ensuring exercise effectiveness, detecting early signs of neurological issues, and customizing rehabilitation programs. By introducing the capability to interpret and ground motion sequences in a spatio-temporal context, MoChat has not just contributed to the development of MLLMs but has also initiated a novel direction for research and practical use in medical motion understanding.

REFERENCES

[1] A. Marusic, S. M. Nguyen, and A. Tapus, "Skeleton-based transformer for classification of errors and better feedback in low back pain physical rehabilitation exercises," *arXiv preprint arXiv:2504.13866*, 2025.

[2] N. Martinel, M. Serrao, and C. Micheloni, "Skelmamba: A state space model for efficient skeleton action recognition of neurological disorders," *arXiv preprint arXiv:2411.19544*, 2024.

[3] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millicah, M. Reynolds, R. Ring, E. Rutherford, S. Cabi, T. Han, Z. Gong, S. Samangooei, M. Monteiro, J. Menick, S. Borgeaud, A. Brock, A. Nematzadeh, S. Sharifzadeh, M. Binkowski, R. Barreira, O. Vinyals, A. Zisserman, and K. Simonyan, "Flamingo: a visual language model for few-shot learning," in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, ser. NIPS '22. Red Hook, NY, USA: Curran Associates Inc., 2024.

[4] OpenAI, "Gpt-4 technical report," 2024. [Online]. Available: https://arxiv.org/abs/2303.08774

[5] W. Hong, W. Wang, Q. Lv, J. Xu, W. Yu, J. Ji, Y. Wang, Z. Wang, Y. Dong, M. Ding, and J. Tang, "Cogagent: A visual language model for gui agents," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 14 281–14 290.

[6] K. Li, Y. He, Y. Wang, Y. Li, W. Wang, P. Luo, Y. Wang, L. Wang, and Y. Qiao, "Videochat: Chat-centric video understanding," 2024. [Online]. Available: https://arxiv.org/abs/2305.06355

[7] J. Li, D. Li, S. Savarese, and S. Hoi, "BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," in *Proceedings of the 40th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., vol. 202. PMLR, 23–29 Jul 2023, pp. 19 730–19 742. [Online]. Available: https://proceedings.mlr.press/v202/li23q.html

[8] C. Guo, X. Zuo, S. Wang, and L. Cheng, "Tm2t: Stochastic and tokenized modeling for the reciprocal generation of 3d human motions and texts," in *Computer Vision – ECCV 2022*, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds. Cham: Springer Nature Switzerland, 2022, pp. 580–597.

[9] B. Jiang, X. Chen, W. Liu, J. Yu, G. Yu, and T. Chen, "Motiongpt: Human motion as a foreign language," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[10] B. Zhu, B. Lin, M. Ning, Y. Yan, J. Cui, W. HongFa, Y. Pang, W. Jiang, J. Zhang, Z. Li, C. W. Zhang, Z. Li, W. Liu, and L. Yuan, "Languagebind: Extending video-language pretraining to n-modality by language-based semantic alignment," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=QmZKc7UZCy

[11] H. Yan, Y. Liu, Y. Wei, G. Li, and L. Lin, "Skeletonmae: Graph-based masked autoencoder for skeleton sequence pre-training," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[12] L. Huang, Y. Huang, W. Ouyang, and L. Wang, "Part-level graph convolutional network for skeleton-based action recognition," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, pp. 11 045–11 052, Apr. 2020. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/6759

[13] S. Ren, L. Yao, S. Li, X. Sun, and L. Hou, "Timechat: A time-sensitive multimodal large language model for long video understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 14 313–14 323.

[14] L. Qian, J. Li, Y. Wu, Y. Ye, H. Fei, T.-S. Chua, Y. Zhuang, and S. Tang, "Momentor: Advancing video large language model with fine-grained temporal reasoning," in *Proceedings of the 41st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, J. Scarlett, and F. Berkenkamp, Eds., vol. 235. PMLR, 21–27 Jul 2024, pp. 41 340–41 356. [Online]. Available: https://proceedings.mlr.press/v235/qian24a.html

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[16] C. Guo, S. Zou, X. Zuo, S. Wang, W. Ji, X. Li, and L. Cheng, "Generating diverse and natural 3d human motions from text," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 5152–5161.

[17] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-Stream Adaptive Graph Convolutional Networks for Skeleton-Based Action Recognition," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA: IEEE, Jun. 2019, pp. 12 018–12 027.

[18] H. Duan, Y. Zhao, K. Chen, D. Lin, and B. Dai, "Revisiting Skeleton-Based Action Recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 2969–2978.

[19] Y. Chen, Z. Zhang, C. Yuan, B. Li, Y. Deng, and W. Hu, "Channel-wise Topology Refinement Graph Convolution for Skeleton-Based Action Recognition," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021, pp. 13 339–13 348.

[20] S. Yan, Y. Xiong, and D. Lin, "Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018.

[21] T. Guo, H. Liu, Z. Chen, M. Liu, T. Wang, and R. Ding, "Contrastive Learning from Extremely Augmented Skeleton Sequences for Self-Supervised Action Recognition," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 1, pp. 762–770, Jun. 2022.

[22] Y. Chen, L. Zhao, J. Yuan, Y. Tian, Z. Xia, S. Geng, L. Han, and D. N. Metaxas, "Hierarchically Self-supervised Transformer for Human Skeleton Representation Learning," in *Computer Vision – ECCV 2022*, ser. Lecture Notes in Computer Science, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds. Cham: Springer Nature Switzerland, 2022, pp. 185–202.

[23] B. Kim, H. J. Chang, J. Kim, and J. Y. Choi, "Global-local motion transformer for unsupervised skeleton-based action learning," in *Computer Vision – ECCV 2022*, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds. Cham: Springer Nature Switzerland, 2022, pp. 209–225.

[24] A. Van Den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.

[25] Z. Zhou, Y. Wan, and B. Wang, "Avatargpt: All-in-one framework for motion understanding planning generation and beyond," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 1357–1366.

[26] H. Liu, C. Li, Y. Li, and Y. J. Lee, "Improved baselines with visual instruction tuning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 26 296–26 306.

[27] P. Jin, R. Takanobu, W. Zhang, X. Cao, and L. Yuan, "Chat-univi: Unified visual representation empowers large language models with image and video understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 13 700–13 710.

[28] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[29] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:231591445

[30] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, I. Stoica, and E. P. Xing, "Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality," March 2023. [Online]. Available: https://lmsys.org/blog/2023-03-30-vicuna/

[31] W. Wang, Q. Lv, W. Yu, W. Hong, J. Qi, Y. Wang, J. Ji, Z. Yang, L. Zhao, X. Song, J. Xu, B. Xu, J. Li, Y. Dong, M. Ding, and J. Tang, "Cogvlm: Visual expert for pretrained language models," 2024. [Online]. Available: https://arxiv.org/abs/2311.03079

[32] W. Dai, J. Li, D. Li, A. Tiong, J. Zhao, W. Wang, B. Li, P. Fung, and S. Hoi, "InstructBLIP: Towards general-purpose vision-language models with instruction tuning," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. [Online]. Available: https://openreview.net/forum?id=vvoWPYqZJA

[33] J. Zhang, C. Herrmann, J. Hur, E. Chen, V. Jampani, D. Sun, and M.-H. Yang, "Telling left from right: Identifying geometry-aware semantic correspondence," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 3076–3085.

[34] H. Cui and T. Hayama, "Joint-partition group attention for skeleton-based action recognition," *Signal Processing*, vol. 224, p. 109592, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0165168424002111

[35] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-iou loss: Faster and better learning for bounding box regression," in *The AAAI Conference on Artificial Intelligence (AAAI)*, 2020, pp. 12 993–13 000.

[36] T. GLM, A. Zeng, B. Xu, B. Wang, C. Zhang, D. Yin, D. Rojas, G. Feng, H. Zhao, H. Lai, H. Yu, H. Wang, J. Sun, J. Zhang, J. Cheng, J. Gui, J. Tang, J. Zhang, J. Li, L. Zhao, L. Wu, L. Zhong, M. Liu, M. Huang, P. Zhang, Q. Zheng, R. Lu, S. Duan, S. Zhang, S. Cao, S. Yang, W. L. Tam, W. Zhao, X. Liu, X. Xia, X. Zhang, X. Gu, X. Lv, X. Liu, X. Liu, X. Yang, X. Song, X. Zhang, Y. An, Y. Xu, Y. Niu, Y. Yang, Y. Li, Y. Bai, Y. Dong, Z. Qi, Z. Wang, Z. Yang, Z. Du, Z. Hou, and Z. Wang, "Chatglm: A family of large language models from glm-130b to glm-4 all tools," 2024.

[37] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "SMPL: A skinned multi-person linear model," *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, vol. 34, no. 6, pp. 248:1–248:16, Oct. 2015.

[38] M. Plappert, C. Mandery, and T. Asfour, "The kit motion-language dataset," *Big Data*, vol. 4, no. 4, pp. 236–252, 2016, pMID: 27992262. [Online]. Available: https://doi.org/10.1089/big.2016.0028

[39] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the*

40th Annual Meeting on Association for Computational Linguistics, ser. ACL '02. USA: Association for Computational Linguistics, 2002, p. 311–318. [Online]. Available: https://doi.org/10.3115/1073083.1073135

[40] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 74–81. [Online]. Available: https://aclanthology.org/W04-1013

[41] R. Vedantam, C. Lawrence Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[42] T. Zhang*, V. Kishore*, F. Wu*, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=SkeHuCVFDr

[43] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica, "Judging LLM-as-a-judge with MT-bench and chatbot arena," in *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. [Online]. Available: https://openreview.net/forum?id=uccHPGDlao

[44] X. Ding, J. Han, H. Xu, W. Zhang, and X. Li, "Hilm-d: Enhancing mllms with multi-scale high-resolution details for autonomous driving," 2025. [Online]. Available: https://arxiv.org/abs/2309.05186

[45] B. Malin, T. Kalganova, and N. Boulgouris, "A review of faithfulness metrics for hallucination assessment in large language models," *arXiv preprint arXiv:2501.00269*, 2024.