CapeLLM: Support-Free Category-Agnostic Pose Estimation with Multimodal Large Language Models

Junho Kim Hyungjin Chung* Byung-Hoon Kim*
EverEx
{jh.kim,hj.chung,bh.kim}@everex.co.kr

Abstract

Category-agnostic pose estimation (CAPE) has traditionally relied on support images with annotated keypoints. This process is often cumbersome and may fail to fully capture the necessary correspondences across diverse object categories. Recent efforts have explored the use of text queries, leveraging their enhanced stability and generalization capabilities. However, existing approaches still fall short in their versatility and expressivity, with dependence on additional support queries, suboptimal use of language priors, and simplistic parametric distributions. To address these limitations, we introduce CapeLLM, the first multimodal large language model (MLLM) designed for CAPE. Our method is completely support-free, necessitating only the detailed text description of the keypoint, along with the query image. For seamless adoption of MLLM to CAPE, we propose effective training strategies and carefully designed instructions, along with inference mechanisms to enhance the visual reasoning process for unseen keypoints. Furthermore, naturally due to the design, CapeLLM is capable of modeling the underlying spatial distribution and uncertainty, allowing for adaptive refinement based on contextual cues. Above all the advantages, we set the new state-of-the-art on the MP-100 benchmark, surpassing 5-shot settings of previous art even in the 1-shot setting.

1. Introduction

Research in pose estimation has evolved independently within distinct domains, with approaches tailored to specific categories such as humans [24, 31, 34], vehicles [17, 18], and animals [10, 35]. Category-Agnostic Pose Estimation (CAPE) extends this task to multiple categories, predicting keypoint positions of novel objects by employing the existing input image (called "query image") with additional information, such as support images and their corresponding keypoint annotations that belong to the same category as the query but has a different pose from it.

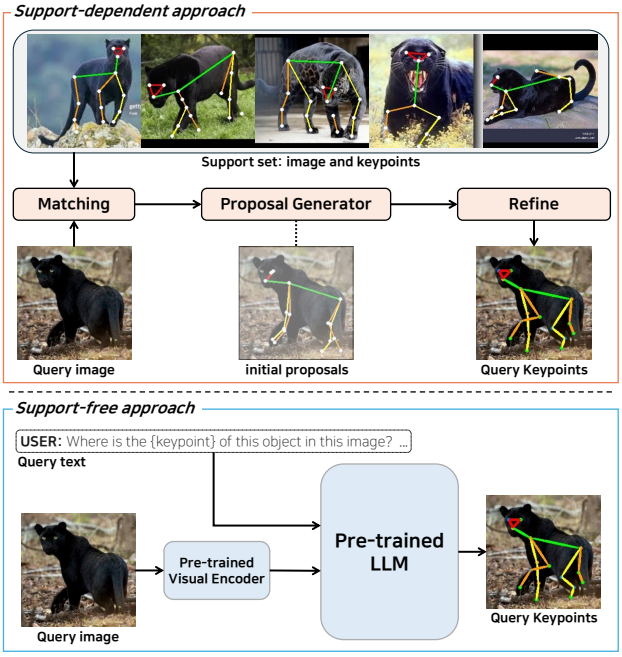

Figure 1. (**Top**) Previous methods that are support-dependent require support images and keypoint annotations. (**Bottom**) Our support-free approach does not need any additional images and annotations, but just a text description of the keypoints. The connectivity between keypoints is pre-defined, which remains consistent across all following figures.

Most existing CAPE methods leverage the support data and adopt a two-stage architecture [22], maximizing the similarity between the query and the support features, with additional refinement steps to improve performance. However, the process of preparing additional queries along with their annotations is impractical and cumbersome, and the performance varies depending on the support set chosen. While CapeX [20] incorporated a text-based approach to mitigate this issue, it still heavily relies on the skeletal representations that are provided as auxiliary information.

To overcome these drawbacks, in this work, we propose

*Corresponding authors

CapeLLM, the first Multimodal Large Language Model (MLLM) for CAPE that is completely support-free. See Figure 1 for the conceptual illustration of CapeLLM against previous support-dependent approaches. Our contributions can be summarized as follows:

- We propose CapeLLM, the first support-free framework in CAPE with advanced query-text comprehension capabilities leveraging an MLLM.
- We elucidate the design choices of using MLLMs for CAPE, from the design of the user query to specific training strategies. Interestingly, we reveal that tailored instructions are the key to unleashing the capabilities of MLLMs in CAPE.
- We propose dynamic round training, enabling spatial reasoning across multiple target poses.
- We propose a flexible decoding strategy that can implicitly model a general probability distribution over keypoints, rather than a fixed parametric model such as a Gaussian
- We achieve state-of-the-art results on the MP-100 benchmark for CAPE, even outperforming the 5-shot accuracy of the previous art [7].

2. CapeLLM

This section is structured as follows. In Sec. 2.1, we describe the model architecture for estimating novel keypoints from given textual descriptions. Then, we present how to design the keypoint names and their corresponding descriptions in Sec. 2.2. Finally, in Sec. 2.3, we introduce two distinct training strategies to endow the spatial reasoning capabilities to CapeLLM.

2.1. CapeLLM Model Architecture

Similar to prior works [13, 25], we utilize a pre-trained visual encoder (e.g., DINO-v2 [15]) in conjunction with a language model (e.g., LLaMA3.1 [6]). The overall architecture is schematically illustrated in Fig. 2. The input image $\mathbf{x} \in \mathbb{R}^{H \times W \times 3}$ is first divided into small patches, where H and W are the height and the width of the image, respectively. The patches are processed through the visual encoder f_ϕ to obtain the patch-processed image features $\tilde{\mathbf{V}} \in \mathbb{R}^{N_v \times C}$, where N_v is the number of patches and C is the dimension of each patch. These patches are linearly transformed into image tokens $\mathbf{V} \in \mathbb{R}^{N_v \times D}$, via a simple learnable matrix

$$\tilde{\mathbf{V}} = f_\phi(\mathbf{x}) \tag{1}$$

$$\mathbf{V} = \tilde{\mathbf{V}}\mathbf{W}_{\text{proj}}, \quad \mathbf{W}_{\text{proj}} \in \mathbb{R}^{C \times D}, \tag{2}$$

where D represents the dimension of each image token.

These image tokens are prepended to the the query text token embeddings $\mathbf{T} \in \mathbb{R}^{N_t \times D}$, where N_t denotes the number of text tokens, and then fed into the language model as the final input tokens $\mathbf{X} \in \mathbb{R}^{N \times D}$:

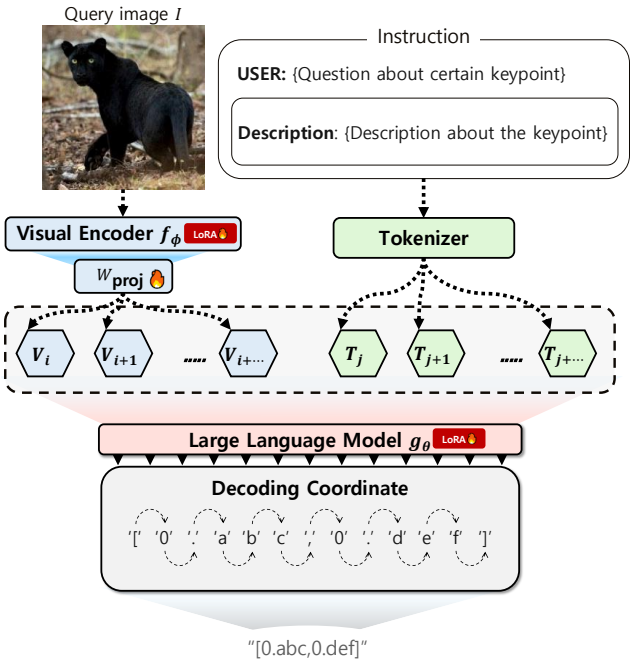

Figure 2. CapeLLM Architecture. CapeLLM consists of two pre-trained modules, visual encoder f_ϕ and LLM g_θ. The visual tokens V from visual encoder f_ϕ are fed into the LLM g_θ with the text tokens T. The decoding strategy predicts the keypoint coordinate directly as text generation.

$$\mathbf{X} = [\mathbf{V}; \mathbf{T}], \tag{3}$$

where ; denotes concatenation of the two matrices along the token dimension, and $N = N_v + N_t$.

The decoder-only LLM g_θ processes the input X to produce the output token matrix $\mathbf{Z}$ which has the same shape as $\mathbf{X}$:

$$\mathbf{Z} = g_\theta(\mathbf{X}). \tag{4}$$

Finally, following previous practices [13, 25], a linear transformation with learnable parameters $\mathbf{W}_{\text{logit}} \in \mathbb{R}^{D \times M}$ outputs the final logits $\mathbf{Y} \in \mathbb{R}^{N \times M}$ for each token, where M is the size of the vocabulary:

$$\mathbf{Y} = \mathbf{Z}\mathbf{W}_{\text{logit}}. \tag{5}$$

While there are different strategies to decode the final predicted output of the keypoint, we choose a strategy that endows maximal flexibility in the output distribution. Specifically, we use a decodable floating-point representation by estimating with the following template: `[0.abc, 0.def]`, as shown in Fig. 2. Here, each decimal point is represented by a separate token.

Concretely, let $y \in \mathbb{R}$ be either of the two scalar values that should be predicted. Then, CapeLLM approximates this value by factorizing it over $K = 3$ digit tokens, i.e.

$$p(y|\mathbf{x}) \approx p_{\phi,\theta}(\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_K|\mathbf{x}) = \prod_{k=1}^{K} p_{\phi,\theta}(\mathbf{Y}_k|\mathbf{x}, \mathbf{Y}_1, \ldots, \mathbf{Y}_{k-1}). \quad (6)$$

Interestingly, one can show that (See Theorem 1 of [23]) CapeLLM models a truncated conditional density up to a resolution of 10^{-K}. This leads to vastly enhanced flexibility, as opposed to the widely used Gaussian parametrization (See Fig. 4). Architectural experiments can be seen in the Supplementary Sec. B.2.

2.2. Design of Instructions

Following the previous clues that instructions exert a significant influence on MLLM's performance [4, 13, 29, 37], we observe that it is insufficient to use only keypoint names to infer their positions within images. Since some categories have densely defined keypoints, especially human faces, we manually design these descriptions to clearly delineate the differences between them. In crafting the descriptions, we intentionally avoid vague or ambiguous phrasing, instead providing precise details about the spatial positions and relationships among keypoints. For instance, when describing the "front wheel" of a swivel chair, rather than saying "starting from this wheel, the remaining wheels are located clockwise," we express it as "next to the fifth wheel and in front of the second wheel". Table 15 illustrates an example of *animal body*. For extensive ablation studies regarding the design of the instructions, see Supplementary Sec B.1.

2.3. Training Strategy

Although the CAPE benchmark dataset, MP-100, covers a wide array of categories, each category contains only around 200 samples on average, which is considerably smaller compared to other benchmarks(MSCOCO [12], MPII [2], AP-10K [35]). The approach of matching an individual image with just a subset of keypoints, as used in LocLLM [25], proves insufficient (See Tab. 2).

Fixed round training Instead, we form (image, keypoints) pairs where every image is paired with *all* of its keypoints during training ("*Fixed-Round*" strategy). Initially, we partition the keypoints (K_{category}) into groups of k. Each set of keypoints is then combined with an image. It is important to note that we permit the repetition of images until every keypoint has been included in a pair. This approach guarantees that no keypoints are left unpaired during the training phase.

Dynamic round training In addition to the Fixed-Round conversation method above, we also explore a "*Dynamic-Round*" strategy. Although both methods involve pairing an image with all the keypoints, the Dynamic-Round approach differs in that the number of keypoints linked to an image varies for each pair, unlike the fixed count of k in the Fixed-Round method. This variability is intended to reinforce the reasoning process by utilizing information from other keypoints during prediction.

3. Experiment

3.1. Experimental Setup

We utilize the MP-100 benchmark dataset [33], aligning with prior methods [3, 7, 14, 19, 20, 22, 33]. MP-100 comprises 100 categories and approximately 20,000 images, split into train, validation, and test sets in a 70/10/20 ratio. We use PCK@0.2 as a measurement of accuracy and compare both quantitative and qualitative results against two representative baselines: GraphCAPE [7] and CapeX [20].

3.2. Main Results

Evaluation dataset	Model	split2	split4	Avg
Support-Query Pairs	GraphCAPE(1-shot) [7]	89.79	87.81	90.52
	GraphCAPE(5-shot) [7]	90.78	90.42	92.04
	CapeX [20]	91.08	89.83	91.62
	CapeLLM (**Ours**)	**92.40**	**90.90**	**92.60**
Only Query Images	CapeX [20]	91.08	89.67	91.59
	CapeLLM (**Ours**)	**92.34**	**90.87**	**91.60**

Table 1. PCK@0.2 on the MP-100 dataset.

Method	Training	Inference		Metric		
		Single	Cumulative	PCK0.15	PCK0.20	PCK0.25
LocLLM-style [25]	-	✓	×	91.00	94.85	96.98
		×	✓	90.56	93.39	95.03
Ours	Fixed	✓	×	95.26	96.98	97.90
		×	✓	93.02	94.59	95.42
	Dynamic	✓	×	94.31	96.05	97.26
		×	✓	**95.62**	**97.28**	**98.27**

Table 2. Result of cumulative reasoning. Default config .

Quantitative results Table 1 shows that CapeLLM outperforms both baselines [7, 20] in all conditions, and sets the new state-of-the-art. This result demonstrates that even without any support, it is possible to attain superior results in CAPE. Moreover, our dynamic-round training makes the model more accurate with our proposed inference mechanism, called "cumulative reasoning", that the prompt employed for predicting each keypoint is prepended to the prompt for the subsequent keypoints, thereby establishing a cumulative context (see the 2nd and 4th row). For the details on cumulative reasoning, refer to Sec. D.

Qualitative results Fig. 3 illustrates that CapeLLM is superior to conventional methods [7, 20] across various cat-

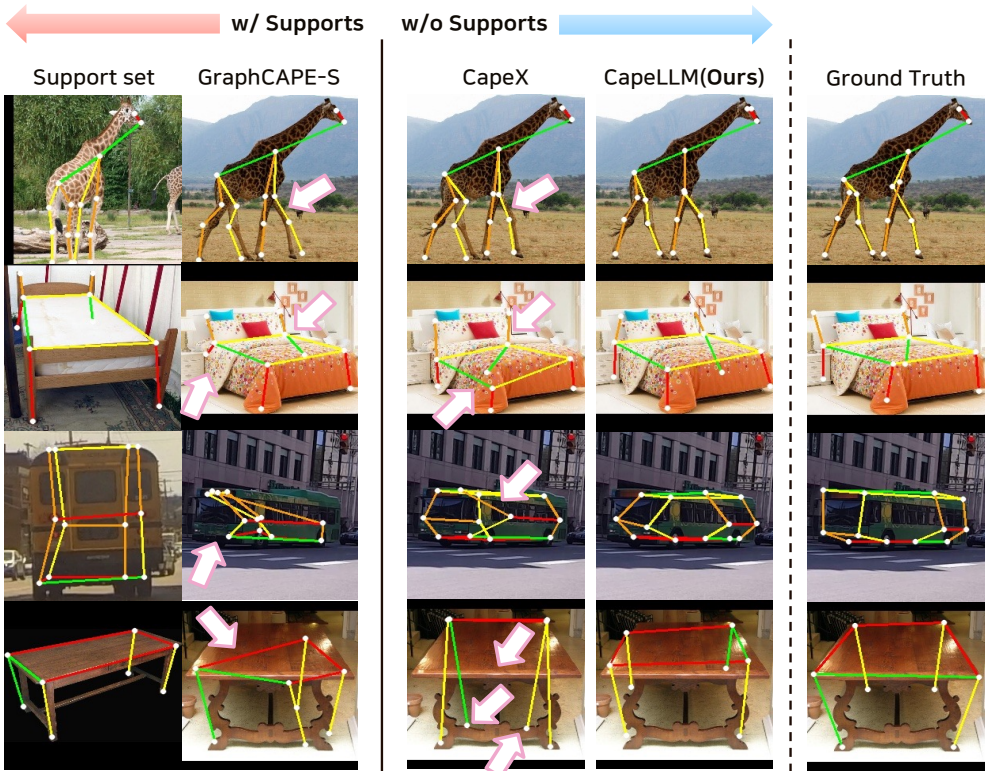

Figure 3. Qualitative results on MP-100. The support set is only used for GraphCAPE [7]

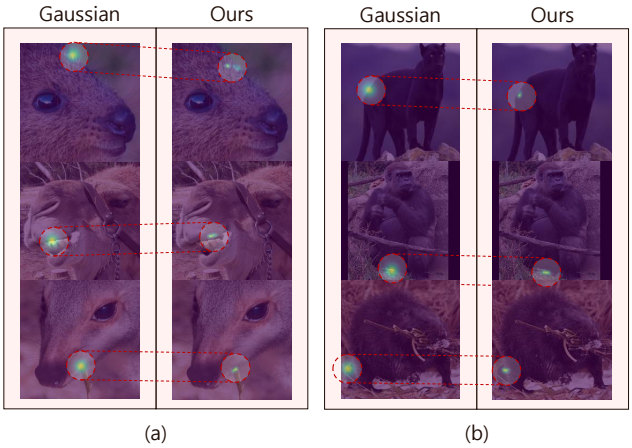

Figure 4. Distribution modeling for keypoints. "Gaussian" displays a fixed-variance Gaussian around the ground-truth target point. "Ours" displays the distribution modeled by CapeLLM, achieved by sampling multiple points, and using kernel density estimation. In (a) **animal face**, (top) *top left side of the left eye*; (mid) *right side of the lip*; (bottom) *left side of the lip*. In (b) **animal body**, (top) *root of tail*; (mid) *right back paw*; (bottom) *left knee*.

egories. Moreover, in Fig. 4, the output coordinate distribution of CapeLLM is kept within the foreground while the distribution of prevalent Gaussian modeling stretches across the background. This observation implies that our decoding-based modeling can be another option to build a density function of keypoint and resolve the inherent constraints in the conventional method (e.g., fixed-variance Gaussian).

4. Conclusion

We introduce CapeLLM, the first fully support-free MLLM-based method for CAPE. By leveraging the reasoning capabilities of a large language model (LLM), CapeLLM achieves state-of-the-art performance without requiring any support images or annotations. We design structured keypoint instructions to fully harness MLLMs for CAPE, and introduce two novel training strategies—fixed/dynamic-round training—which not only improve category-agnostic keypoint prediction but also enhance cumulative reasoning, allowing the model to refine its predictions iteratively based on contextual information. We believe CapeLLM serves as a foundational step toward the broader application of MLLMs in spatial reasoning and structured perception tasks. Future work can explore more advanced decoding strategies, multi-modal extensions, and scalability to real-world scenarios, paving the way for next-generation keypoint estimation models driven by MLLMs.

References

[1] Meta AI. Llama 3.2: Vision Edge for Mobile Devices, 2024. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/. 8, 9

[2] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 3

[3] Junjie Chen, Jiebin Yan, Yuming Fang, and Li Niu. Meta-Point Learning and Refining for Category-Agnostic Pose Estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23534–23543, 2024. 3

[4] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 3

[5] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision Transformers Need Registers. In *The Twelfth International Conference on Learning Representations*, 2024. 8

[6] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 2, 8, 9

[7] Or Hirschorn and Shai Avidan. A graph-based approach for category-agnostic pose estimation, 2024. 2, 3, 4

[8] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*, 2022. 8

[9] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023. 8, 9

[10] Rollyn Labuguen, Jumpei Matsumoto, Salvador Blanco Negrete, Hiroshi Nishimaru, Hisao Nishijo, Masahiko Takada, Yasuhiro Go, Ken-ichi Inoue, and Tomohiro Shibata. MacaquePose: a novel "in the wild" macaque monkey pose dataset for markerless motion capture. *Frontiers in behavioral neuroscience*, 14:581154, 2021. 1

[11] Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. Lisa: Reasoning segmentation via large language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9579–9589, 2024. 9

[12] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common Objects in Context, 2015. 3

[13] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024. 2, 3, 7, 8, 9

[14] Khoi Duc Nguyen, Chen Li, and Gim Hee Lee. ESCAPE: Encoding Super-keypoints for Category-Agnostic Pose Estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23491–23500, 2024. 3

[15] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research*, 2024. 2, 8

[16] Kanchana Ranasinghe, Satya Narayan Shukla, Omid Poursaeed, Michael S Ryoo, and Tsung-Yu Lin. Learning to localize objects improves spatial reasoning in visual-llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12977–12987, 2024. 9

[17] N Dinesh Reddy, Minh Vo, and Srinivasa G Narasimhan. Carfusion: Combining point tracking and part detection for dynamic 3d reconstruction of vehicles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1906–1915, 2018. 1

[18] N Dinesh Reddy, Minh Vo, and Srinivasa G Narasimhan. Occlusion-net: 2d/3d occluded keypoint localization using graph networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7326–7335, 2019. 1

[19] Pengfei Ren, Yuanyuan Gao, Haifeng Sun, Qi Qi, Jingyu Wang, and Jianxin Liao. Dynamic Support Information Mining for Category-Agnostic Pose Estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1921–1930, 2024. 3

[20] Matan Rusanovsky, Or Hirschorn, and Shai Avidan. CapeX: Category-Agnostic Pose Estimation from Textual Point Explanation. *arXiv preprint arXiv:2406.00384*, 2024. 1, 3

[21] Chaitanya Ryali, Yuan-Ting Hu, Daniel Bolya, Chen Wei, Haoqi Fan, Po-Yao Huang, Vaibhav Aggarwal, Arkabandhu Chowdhury, Omid Poursaeed, Judy Hoffman, et al. Hiera: A hierarchical vision transformer without the bells-and-whistles. In *International Conference on Machine Learning*, pages 29441–29454. PMLR, 2023. 8

[22] Min Shi, Zihao Huang, Xianzheng Ma, Xiaowei Hu, and Zhiguo Cao. Matching is not enough: A two-stage framework for category-agnostic pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7308–7317, 2023. 1, 3

[23] Xingyou Song and Dara Bahri. Decoding-based regression. *arXiv preprint arXiv:2501.19383*, 2025. 3

[24] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE/CVF conference on*

computer vision and pattern recognition, pages 5693–5703, 2019. 1

[25] Dongkai Wang, Shiyu Xuan, and Shiliang Zhang. LocLLM: Exploiting Generalizable Human Keypoint Localization via Large Language Model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 614–623, 2024. 2, 3, 8, 9

[26] Wenhai Wang, Zhe Chen, Xiaokang Chen, Jiannan Wu, Xizhou Zhu, Gang Zeng, Ping Luo, Tong Lu, Jie Zhou, Yu Qiao, et al. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks. *Advances in Neural Information Processing Systems*, 36, 2024. 8, 9

[27] Fei Wei, Xinyu Zhang, Ailing Zhang, Bo Zhang, and Xiangxiang Chu. Lenna: Language enhanced reasoning detection assistant. *arXiv preprint arXiv:2312.02433*, 2023. 9

[28] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. 9

[29] Junda Wu, Xintong Li, Tong Yu, Yu Wang, Xiang Chen, Jiuxiang Gu, Lina Yao, Jingbo Shang, and Julian McAuley. Commit: Coordinated instruction tuning for multimodal large language models. *arXiv preprint arXiv:2407.20454*, 2024. 3

[30] Jiannan Wu, Muyan Zhong, Sen Xing, Zeqiang Lai, Zhaoyang Liu, Zhe Chen, Wenhai Wang, Xizhou Zhu, Lewei Lu, Tong Lu, Ping Luo, Yu Qiao, and Jifeng Dai. VisionLLM v2: An End-to-End Generalist Multimodal Large Language Model for Hundreds of Vision-Language Tasks. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 9

[31] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 466–481, 2018. 1

[32] Guowei Xu, Peng Jin, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. LLaVA-CoT: Let Vision Language Models Reason Step-by-Step, 2025. 9

[33] Lumin Xu, Sheng Jin, Wang Zeng, Wentao Liu, Chen Qian, Wanli Ouyang, Ping Luo, and Xiaogang Wang. Pose for everything: Towards category-agnostic pose estimation. In *European conference on computer vision*, pages 398–416. Springer, 2022. 3

[34] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. Vitpose: Simple vision transformer baselines for human pose estimation. *Advances in Neural Information Processing Systems*, 35:38571–38584, 2022. 1

[35] Hang Yu, Yufei Xu, Jing Zhang, Wei Zhao, Ziyu Guan, and Dacheng Tao. AP-10K: A Benchmark for Animal Pose Estimation in the Wild. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. 1, 3

[36] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023. 8, 9

[37] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. MiniGPT-4: Enhancing Vision-Language Understanding with Advanced Large Language Models. In *The Twelfth International Conference on Learning Representations*, 2024. 3, 8

CapeLLM: Support-Free Category-Agnostic Pose Estimation with Multimodal Large Language Models

Supplementary Material

A. Keypoint Descriptions

We create the names and descriptions of keypoints for all 100 categories. The names can be divided into two types: one that has its own unique name, e.g., `left shoulder`, `right eye`, and the other that does not have its own name. the latter is difficult to define due to the densely distributed position. We concentrate on designating the latter and determine the names using their relative positions in each category; for example, "upper", "central", "lower". The descriptions are represented with the keypoint position in the category and its relation with other keypoints; e.g., in the *animal body*, the description of `left front paw` is defined as "The left front paw is the lower end of the left forelimb, used for movement and manipulation of objects. It is positioned below the left elbow and connected with the left elbow". A detailed example can be found in Table 15.

B. Exploring other Design Choices

B.1. Instruction

w/ description	w/ keypoint list	PCK@0.05	PCK@0.2	mPCK
×	×	72.60	96.22	89.86
✓	×	**78.43**	**96.98**	**91.98**
✓	✓	77.36	95.80	90.97

Table 3. Effect of additional info for keypoints in training. Default config .

Diverse questions	Add conversation outline	PCK@0.05	PCK@0.2	mPCK
×	×	**78.43**	**96.98**	**91.98**
✓	×	74.24	96.56	90.56
×	✓	75.08	96.27	90.63
✓	✓	68.24	95.93	88.52

Table 4. Effect of adding a conversation outline and diversifying question expressions. Default config .

Instruction variations As mentioned in Sec 2.2, we include not only the names but also descriptions of the keypoints in the instructions to help the model better to reason the location of keypoints. We examine how the description affects model performance by training the model without descriptions. The result in Table 3 shows that without descriptions, the accuracy decreases over 2%p in mPCK, suggesting that the keypoint description plays a significant role in enhancing to find the exact position. We experiment another scenario to include all keypoint names for each category in the instruction as "Keypoint List". As shown in Table 3, unlike keypoint descriptions, the list of keypoint names is not helpful for improving the model, rather reducing its performance. Next, we explore whether two optional conditions affect the performance or not: one is encompassing a conversation outline [13] and the other is to diversify the question expression in instruction. The outline slightly modified from the prior work [13] seems not to influence to solve the problem that predicts coordinates, and the random question does not have any positive effect on the performance, actually leading to a decrease in the model's performance(Table 4).

Description Type	PCK@0.05	PCK@0.2	mPCK
Vague	69.82	91.97	85.98
Spatial & Relational	**78.43**	**96.98**	**91.98**

Table 5. Comparison with vague descriptions. Our method .

Vague description To investigate the impact of instructions on the performance, we define two types of variations in the instruction: (1) change in the keypoint names and descriptions, (2) the omission of keypoint description. Initially, we consider a scenario in which vague keypoint descriptions are provided, as mentioned in Sec. 2.2, and convert descriptions with the ambiguous ones in training. We find that incorporating detailed spatial and relational information among keypoints yields an improvement of up to 6%p in mPCK compared to the baseline (Table 5). This finding highlights the critical role of learning spatial positioning and contextual relationships for accurately predicting keypoint coordinates.

Name	Description	PCK0.05	PCK0.20	PCK0.25
×	×	**78.43**	**96.98**	**97.90**
✓	×	78.24	96.94	97.83
×	✓	70.11	96.67	97.69
✓	✓	66.16	96.09	97.26

Table 6. Robustness to variation in input. Default config .

Different style of description We also assess the model's stability when confronted with input styles that differ from those seen during training. By utilizing GPT-4o, we prompt it to convert the keypoint names and descriptions in the test set into a simplified format that preserves their original meaning. According to Table 6, CapeLLM exhibits notably consistent performance across these variations.

Without description We scrutinize two cases where the descriptions are excluded: (1) employing a simple QA

In Training		In Inference		Metric	
Random Replaced	Detail Desc	Replaced	Removed	PCK0.20	mPCK
×	✓	×	×	**96.98**	**91.98**
×	×	✓	×	95.28	86.66
×	×	×	✓	95.15	85.69
✓	✓	×	×	96.65	89.87
✓	×	✓	×	96.39	89.65
✓	×	×	✓	96.07	88.84

Table 7. Performance with varying descriptions. Default config .

method to query keypoint coordinates without any description, (2) where the description is replaced with the statement "There is no description to refer to.". In all cases, CapeLLM maintains relatively stable performance, and its robustness can be further enhanced when taking advantage of the random substitution of descriptions during training (Table 7).

Multi-round	PCK0.05	PCK0.10	PCK0.15	PCK0.20	PCK0.25	mPCK
$k = 1$	78.29	91.55	95.19	96.89	97.88	91.96
$k = 2$	72.82	88.06	92.79	95.30	96.56	89.11
$k = 4$	**78.43**	**91.34**	**95.26**	**96.98**	**97.90**	**91.98**
$k = 6$	74.33	89.82	94.17	96.36	97.46	90.43
$k = 8$	75.28	89.89	93.99	96.16	97.41	90.55

Table 8. Ablation in multi-round k. Default config .

Choice of round k We investigate the optimal number of rounds k in the conversation. Table 8 shows that under the same training conditions, the highest performance was observed when k is set to 4. No explicit tendency was found as k changed.

LLM	Step-by-step instruction	PCK@0.05	PCK@0.2	mPCK
Llama3.1-8B [6]	×	**78.43**	**96.98**	**91.98**
	✓	76.06	96.48	91.11
Llama3.2-1B [1]	×	76.46	96.41	91.20
	✓	76.65	96.75	91.49

Table 9. Performance comparison with *step-by-step instruction* across different LLMs. Default config .

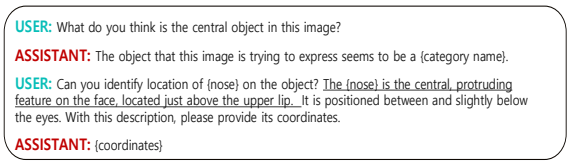

Figure 5. Step-by-step instruction. The nose is in the example above, which can be replaced with whatever you want to find out. The underlined is the description of nose, which can also be replaced according to the keypoint.

Different style of instruction We take another structure of instruction question-answering in a step-by-step manner, so-called *step-by-step instruction*(Figure 5). Specifically, Rather than providing instruction as Figure 2, we question what the object is and then inquire the coordinates of keypoints. We expect this approach would help the model better understand the input. Interestingly, the effect of this mechanism varies depending on the LLM, as in Table 9. It appears that different LLMs require different approaches to better understand the instruction.

B.2. Architecture

Visual Encoder	PCK0.05	PCK0.10	PCK0.15	PCK0.20	PCK0.25	mPCK
DINO-v2-reg [5]	62.52	86.00	92.83	95.83	97.34	86.90
Hiera [21]	56.13	83.31	91.99	95.67	97.35	84.89
DINO-v2 [15]	**78.43**	**91.34**	**95.26**	**96.98**	**97.90**	**91.98**

Table 10. Ablation in visual encoders. Default config .

Fine-tuning method	PCK0.05	PCK0.10	PCK0.15	PCK0.20	PCK0.25	mPCK
None (Frozen)	69.69	88.16	92.62	95.07	96.41	88.39
LoRA [8]	**78.43**	**91.34**	**95.26**	**96.98**	**97.90**	**91.98**
Full parameters	6.93	23.72	42.41	55.31	64.56	38.59

Table 11. Ablation in fine-tuning methods. Default config .

Choice of visual encoder We conduct an ablation study for the visual encoder in CapeLLM. We choose three popular visual encoders: DINO-v2 [15], Hiera [21], DINO-v2-reg [5], which are pre-trained on same dataset. Table 10 shows that using DINO-v2 [15] yields the highest performance. The known issue in DINO-v2, artifacts in the feature maps [5]), seems to have little impact on performance in the CAPE task. A noteworthy point is the number of image tokens. Although Hiera [21] has 20% less image tokens than the other two encoders, the performance gap is just about 1%p, implying that retaining a larger number of image tokens does not necessarily have something to do with performance. Then, we examine three types of fine-tuning methods: full fine-tuning, fine-tuning with LoRA [8], and freezing. In constrat with the traditional MLLMs [13, 26, 37], visual encoder with LoRA was more advantageous than the other two options as [25](Table 11). Notably, the full fine-tuning approach, where all parameters are learnable, drastically deteriorate the performance. This fact seems to imply that when using relatively small datasets, leaving all parameters trainable may lead to overfitting, thus resulting in severe degradation in performance.

LLM	PCK0.05	PCK0.10	PCK0.15	PCK0.20	PCK0.25	mPCK
Llama3.2-1B [1]	76.46	91.05	94.69	96.41	97.40	91.20
Vicuna-7B-v1.5 [36]	62.15	84.40	91.51	94.79	96.33	85.84
Mistral-7B-v0.3 [9]	77.63	91.32	94.90	96.46	97.54	91.57
Llama3.1-8B [6]	**78.43**	**91.34**	**95.26**	**96.98**	**97.90**	**91.98**

Table 12. Ablation in LLM. Default config .

Choice of LLM To analyze the performance variations coming from different LLMs, we select four most recent

and popular language models: Vicuna-7B [36], Mistral-7B [9], Llama3.1-8B [6], and Llama3.2-1B [1]. We find that the overall accuracy gets improved as the size of the LLM increases(Table 12). Exceptionally, Llama3.2-1B [1] exhibits an overwhelming result surpassing that of a 7B-sized LLM, Vicuna-7B-v1.5, which appears to be the effect of effectively transferring the knowledge of a larger model through distillation training methods [1]. A larger vocabulary size seems to play a essential role to positively influence the integration of visual information and language.

Instruction	Output format	PCK@0.05	PCK@0.2	mPCK
Base instruction	text	**78.43**	**96.98**	**91.98**
	special token	76.06	96.48	91.11
Step-by-step instruction	text	76.46	96.41	91.20
	special token	76.65	96.75	91.49

Table 13. Comparison with token output format. Default config .

Token output format We explore a method that utilizes token embeddings <KEYPOINT> instead of text-based outputs. To introduce this method to our pipeline, some modifications in instruction should be made: the coordinates are replaced with special token <KEYPOINT> as answers, accordingly the vocabulary size increases, and input embeddings are turned into the trainables. The tokens are turned into the output embeddings from the LLM and are fed into a task-specific decoder. Typically, while a grounding-based pre-trained decoder is used in some tasks [11, 27, 30], no suitable decoders exist for CAPE. So, we create a simple decoder that transforms the embeddings into the coordinates and train it from scratch. We validate this method on both default instruction(as Figure 2) and step-by-step one(Figure 5). Despite the lack of pre-training, the method using <KEYPOINT> outputs comparable result to models with default architecture(Table 13).

Pre-training method	PCK@0.05	PCK@0.2	mPCK
w/o pre-training	**78.43**	**96.98**	**91.98**
Direct QA	78.98	96.60	91.96
Step-by-step QA	78.05	96.23	91.40

Table 14. Comparison in pre-training methods. Default config .

C. Pre-Training Strategy

USER: What is the name of a keypoint at [0.123,0.456] in the image? Tell me the name and why.

ASSISTANT: The keypoint name is nose, because the nose is the central, protruding feature on the face, located just above the upper lip. It is positioned between and slightly below the eyes.

Figure 6. Instruction of direct QA for pre-training.

We attempt two types of pre-training process: *direct QA* and *step-by-step QA*. The *direct QA* has an instruction that it is in the form of asking and answering the name of the keypoint corresponding to the coordinates, as in Figure 6. On the other hand, step-by-step QA in Figure 7 has an instruction that is in the form of asking about the category, inquiring the existence of the keypoint in the image, and then inducing the selection of the keypoint corresponding to the coordinates. Referring to the related works [16, 26, 30], all layers except for projection layer are frozen in this stage. As a consequence, there is no positive effect on the performance gain, as shown in Table 14. In light of the use of large-scale pre-training data in the previous methods [16, 25, 26, 30], we conjecture that the limited number of images in each category might result in this outcome.

Joint list: nose, left eye, right eye, ...

USER: What do you think is the central object in this image?

ASSISTANT: The object that this image is trying to express seems to be a {human body}.

USER: What joints can you identify on the {human body} in the image? Answer according to the order in the joint list.

ASSISTANT: The {human body} in the image seems to have several joints in the list, named {nose, left eye, right eye}. Those joints are visible in the image.

USER: Can you identify which joint is located at the coordinate {location}? This coordinate represent relative position within the image, where the top-left corner is defined as [0, 0] and the bottom-right corner as [1, 1]. For example, [0.5, 0.5] indicates the center of the image

ASSISTANT: the joint located at coordinates is {left eye}.

Figure 7. Instruction of step-by-step QA for pre-training.

D. Cumulative Reasoning for Pose Estimation

Inspired by the previous works related to Chain-of-Thought (CoT) [13, 28, 32], we investigate whether the information from certain keypoints influences the estimation of others. Specifically, we devise an inference mechanism, called "*Cumulative Reasoning*", for helping to predict keypoint coordinates more precisely using the implicit capability of MLLMs, and analyze its effectiveness comparing with default process, single-round inference("Single" in Table 2). For this reasoning task, the prompt employed for predicting each keypoint is prepended to the prompt for the subsequent keypoints, thereby establishing a cumulative context. As detailed in Table 2, while the LocLLM [25]-style and the fixed-training scheme result in decreased performance (see the 2nd and 4th rows), dynamic-round training strategy not only encourages the model to achieve better results than single-round inference by 1.2%p (see the last row), but also outperforms the fixed-round strategy in row 3rd and 5th of Table 2. This experiment suggests that CapeLLM can extract richer spatial and relational cues based on the accumulated information from other keypoints, which in turn enables it to reason about the target keypoint's position.

Keypoint	Description
Left eye	The left eye is one of the two visual organs located on the face. It is positioned slightly to the left of the nose and just below the brow ridge, visible from the front.
Right eye	The right eye is the visual organ located on the right side of the face. It is situated to the right of the nose and directly opposite the left eye.
Nose	The nose is the central, protruding feature on the face, located just above the upper lip. It is positioned between and slightly below the eyes
Neck	The neck is the part of the body connecting the head to the torso that refers to the area from the shoulders to the hip joints. It is located below the head, near the junction where the shoulders meet the body.
Root of tail	The root of the tail is at the base of the spine, where the tail begins. It is located near the lower back, above the hips.
Left shoulder	The left shoulder is the joint connecting the left arm to the torso. It is situated to the left of the neck and above the left elbow.
Left elbow	The left elbow is the joint in the middle of the left arm, connecting the upper arm to the forearm. It is located between the left shoulder and the left front paw and connectd with them.
Left front paw	The left front paw is the lower end of the left forelimb, used for movement and manipulation of objects. It is positioned below the left elbow and connected with the left elbow.
Right shoulder	The right shoulder is the joint connecting the right arm to the torso. It is located to the right of the neck and above the right elbow.
Right elbow	The right elbow is the joint in the middle of the right arm, connecting the upper arm to the forearm. It is situated between the right shoulder and the right front paw and connectd with them.
Right front paw	The right front paw is the lower end of the right forelimb, used for movement and manipulation of objects. It is located below the right elbow and connectd with the right elbow.
Left hip	The left hip is the joint connecting the left leg to the torso. It is positioned below the root of the tail and above the left knee.
Left knee	The left knee is the joint in the middle of the left leg, connecting the upper leg to the lower leg. It is located between the left hip and the left back paw and connectd with them..
Left back paw	The left back paw is the lower end of the left hind limb, used for movement and support. It is situated below the left knee.
Right hip	The right hip is the joint connecting the right leg to the torso. It is positioned below the root of the tail and above the right knee.
Right knee	The right knee is the joint in the middle of the right leg, connecting the upper leg to the lower leg. It is located between the right hip and the right back paw and connectd with them.
Right back paw	The right back paw is the lower end of the right hind limb, used for movement and support. It is situated below the right knee.

Table 15. An example of descriptions: *animal body*