Supplementary for Scene-LLM

This document serves as the supplementary material for *Scene-LLM: Extending Language Models for 3D Visual Understanding and Reasoning*. Due to space constraints in the main paper, this supplementary section provides additional experimental results. It detailed benchmark results for scene caption generation, and further ablation studies that explore various modalities, the impact of using frame data, and how camera views and voxel resolution affect performance. We also provide limitation and failure cases stemmed from our method. Additionally, we present comprehensive information on the generation of frame data and scene data, encompassing prompts, post-processing steps, and data distribution specifics. Finally, detailed explanations of the training and inference methodologies as outlined in the main paper are also included for thorough understanding.

Contents

1. Ablation Studies comparing different modalities.

Table 1. Ablation Studies comparing different input modalities. Including textual zero-shot, textual fine-tuned, and textual tasks-specific-tuned, video, bird-eye views, and 3D point sets.

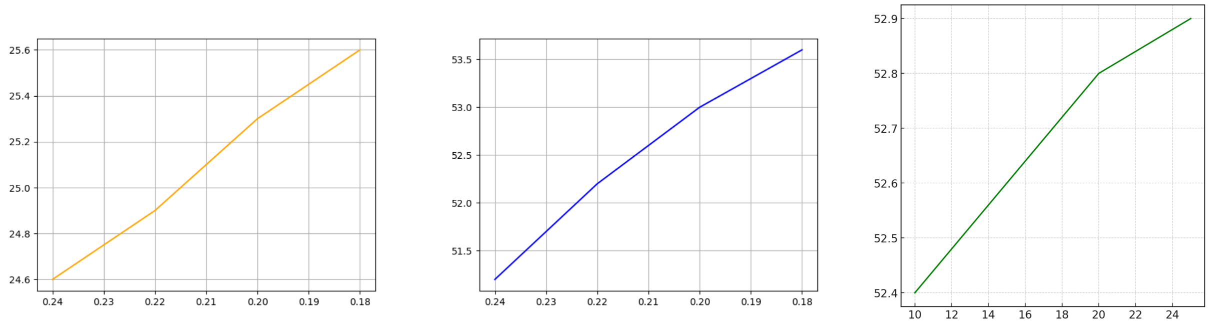
	Mod.	*Tune.*	**ScanQA**	**SQA3D**
(a)	text	Zero-Shot	13.1	34.1
(b)	text	Finetune	17.9	48.9
(c)	text	Task Tune	18.0	49.5
(d)	Video	Task Tune	18.9	48.2
(e)	BEV	Task Tune	19.2	48.4
(f)	3D	Finetune	25.0	52.8

Our study evaluates performance on the ScanQA [2] and SQA3D [7] benchmarks across various modalities, including text, video, Bird's Eye View (BEV), and 3D. In the text modality, we employ the Llama-2-7b [10] backbone, while video and BEV modalities share identical network architectures and training strategies as Scene-LLM. Within the text modality, we compared object bounding boxes and object entities, finding the latter more effective.

We conducted experiments (a-c) using the text modality, encompassing zero-shot inference, fine-tuning with scene data, and additional task-specific tuning. Experiments (d-f) explored video, BEV, and 3D modalities. Our findings reveal that video and BEV modalities outperform the text modality in the ScanQA benchmark but underperform in SQA3D. Both methods underperform 3D data on both benchmarks. This suggests that spatial downsampling methods may not be ideal for processing video and image patches, underscoring the value of 3D information in preserving spatial knowledge more effectively.

2. Ablation Studies comparing different Voxel Grid Resolution.

Due to the max token length of LLama2 is 4096, we use a fixed voxel downsample resolution 0.18 in our experiment. However, in Fig. 1 (a)(b), we show that increasing the resolution result into performance boost on QA benchmarks. This suggest that Scene-LLM might be further improved by incorporating LLMs that process longer text tokens [11, 12] to process dense information more effectively.



(a) ScanQA-Resolution

(b) SQA3D-Resolution

(c) SQA3D-#views

Figure 1. EM scores on the Y-axis with decreasing voxel grid intervals (X-axis), indicating higher resolution leads to better performance. EM score slightly increases with number of views increases.

3. Ablation Studies comparing different Number of views.

During 3D visual feature extracting process, we randomly sample views from input 3D scenes. Fig. 1 (c) reports the Exact Match (EM) score on SQA3D using different number of views. Figure shows the EM score has minor increment with the number of views increases, comparing with the change of voxel resolution. This further shows the main bottleneck for scene feature extraction is the voxel resolution, which is limited by the max token length of LLama2.

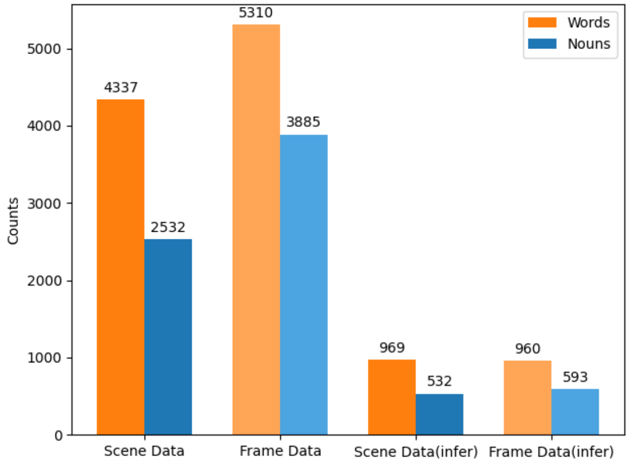


Figure 2. Comparing the number of unique words and nouns in the scene dataset and frame dataset. Comparing the number of unique words and nouns inferred by training with the model with scene data and frame data.

4. Ablation Studies comparing concept numbers.

Fig. 2 presents a detailed analysis comparing the richness of vocabulary and conceptual variety between scene data and frame data, as well as their impact on model training. The analysis focuses on two key metrics: the number of unique words and the number of unique nouns found in the captions of both datasets, excluding stopwords. The scene data, consisting of 200,000 captions, is compared with frame data, which contains 190,000 captions.

In terms of unique words, the metric refers to the total count of distinct words across the dataset, providing insight into the lexical diversity. Similarly, the count of unique nouns, excluding stopwords, gives an indication of the variety and specificity of concepts covered in the data. The figure reveals that frame data exhibits a higher count of both unique words and nouns, suggesting a more diverse and conceptually rich dataset.

Furthermore, the figure also compares the output when the model is trained on these two datasets, using 142 validation scenes from the ScanNet V2 dataset [5]. Notably, the model trained with frame data generates a greater number of nouns during inference, indicating that frame data effectively imparts a broader range of fine-grained concepts to the model. This enhanced conceptual diversity likely contributes to more robust and nuanced model performance in tasks requiring detailed understanding and reasoning.

5. Ablation Studies comparing convergence speed.

Fig. 3 displays a comparative analysis of the training loss trajectories during the first 6,000 steps of the pretraining phase, as specified in [10]. This phase emphasizes concept alignment, leveraging two distinct datasets: frame data, which is accompanied by 190,000 textual annotations, and scene data, with 200,000 textual annotations.

The graph clearly shows a quicker reduction in training loss for the model utilizing frame data as opposed to scene data. This quicker convergence when using frame data suggests that the frame dataset provides a richer and more diverse set of concepts for the model to learn. This diversity likely facilitates more efficient learning and understanding, enabling the model to more rapidly adjust its parameters for optimal performance. Additionally, the swifter convergence with frame data could be linked to the nature of the 3D data itself. Frame point set typically contains more detailed and precise information that aligns closely with its accompanying annotations. In contrast, the scene point set, being a downsampled representation, may not align as closely with its textual annotations. This misalignment could lead to slower learning as the model struggles to correlate the visual data with its textual descriptions, thereby explaining the observed difference in convergence rates.

6. Result on Scene Caption Generation.

We detail our investigation into scene caption generation on the Scan2Cap benchmark [4], using task-specific fine-tuning. This benchmark evaluates the ability of models to generate detailed captions for objects within a 3D scene, based on their bounding box inputs. Similar to the 3D-LLM [6], the model receives the center and dimensions of an object's bounding box as

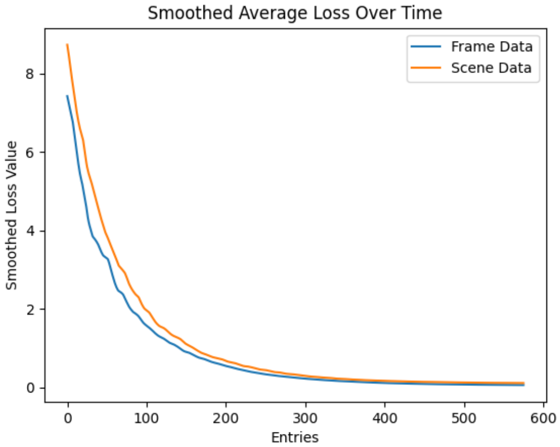


Figure 3. Loss curves for training the projection layer at the first $6k$ iterations. Using frame data converges faster than using scene data.

Table 2. Result on 3D Dense Caption Benchmark Scan2Cap [4].

	CIDEr	**BLEU-4**	**METEOR**	**ROUGE**
Scan2Cap [4]	35.2	22.4	21.4	43.5
3D-LLM [6]	–	8.1	13.1	33.2
Scene-LLM	37.9	24.1	21.8	45.6

input, formatted as `[position x, position y, position z, length x, length y, length z]`, with each dimension represented as a two-digit floating-point number.

In our study, Scene-LLM is compared against both a modular method outlined in [4] and another 3D-visual-language model from [6]. The results demonstrate that Scene-LLM outperforms the other methods across all evaluated metrics, indicating Scene-LLM's proficiency in comprehending 3D scenes and accurately understanding spatial coordinates when fine-tuned for specific tasks. The result indicates that when Scene-LLM is combined with 3D grounding methods, it holds the potential to tackle a wide array of complex tasks.

7. Result on 3DMV-VQA.

Table 3. Result on 3DMV-VQA(single room only) Benchmark.

	Concept	**Counting**	**Relation**	**Comparison**	**Overall**
Scene-LLM(zs)	69.9	31.2	62.0	73.5	59.6
Scene-LLM(tt)	70.2	33.5	62.4	75.1	61.3

We evaluated the performance of Scene-LLM on the 3DMV-VQA benchmark [6], a benchmark specifically designed for testing a model's ability to understand and reason about 3D scenes, using data from the Habitat Matterport 3D Dataset [9].

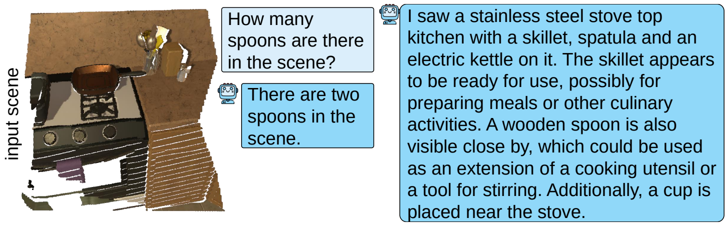
For our evaluation, we selected a subset of the 3DMV-VQA benchmark that aligns with our model's focus on single-room scenarios. This subset, sourced from the benchmark's open-source codebase, comprises a total of $1,212$ scenes. Notably, the answers in this benchmark are generally concise, typically consisting of one word or a short phrase. To align with this format, we employed the identifier `"Answer the question using one word or one phrase"` in our zero-shot (zs) evaluation setup.

We assessed Scene-LLM under two distinct settings: zero-shot and task-specific tuning. In the zero-shot setting, where no additional fine-tuning was applied, Scene-LLM demonstrated reasonable performance, indicating an inherent capability to understand and respond to queries related to 3D scenes. However, when further optimized through task-specific tuning, the model's performance showed a notable improvement.

8. Limitation and Failure Cases

Scene-LLM faces limitations such as LLM input token length, challenges in processing dynamic scenes without a state detector, lacking geometry feature, and language hallucinations.

See Fig. 4 for a failure case, resulted from 1) The limited token length restricts the resolution of input features, causing the model to struggle with counting in cluttered scenes with small objects (three spoons, not one). 2) The Alfred simulator contains less realistic rendering, which can lead the model to extract incorrect semantic features (not cup, but glass container).



How many spoons are there in the scene?

There are two spoons in the scene.

I saw a stainless steel stove top kitchen with a skillet, spatula and an electric kettle on it. The skillet appears to be ready for use, possibly for preparing meals or other culinary activities. A wooden spoon is also visible close by, which could be used as an extension of a cooking utensil or a tool for stirring. Additionally, a cup is placed near the stove.

Figure 4. Failure Case due to limited resolution.

Despite these challenges, Scene-LLM represents an advancement in 3D visual understanding, paving the way for more complex agent interactions in indoor settings.

9. 3D Dataset

Our 3D scene dataset is composed of three parts: we collect 564 training scenes from ScanNet-V2 [5], 1212 scenes from HM3D [9] dataset, and 7075 randomly configued scenes from iThor [1] dataset. There are in total of 8851 scenes. For the 3D frame, we use Habitat simulator and Ai2Thor simulator to collect frame data. We also use the multi-view images from ENet [8] for ScanNet frame data. For each scene, we collect 20 frames from multiple views.

10. Frame Data Generation Details.

10.1. Prompts

We use MiniGPT-V2 [3] for frame data generation. We use the identifier `[grounding]` as we found it can effectively reduce hallucination in the generated texts. Here are the prompts we use:

1. `[grounding] Describe this image in detail. Give as many details as possible. Say everything you see.`

2. `[grounding] Describe the objects in this image.`

The first prompt generates scene captions, and the second prompt generates object-centric descriptions. We filtered out outputs with meaningless characters, and a total of around $190k$ textual descriptions are collected.

11. Scene Data Generation Details.

11.1. Prompts

In the section, we provide the prompts used for scene data generation. The prompts we provided are composed of two parts $P_{context}$ and $P_{instruction}$, where $P_{context}$ introduces the scene context used for data generation and $P_{instruction}$ indicates which type of instructional data to generate.

For the scene context, we include four types of context. Here are the context types and the corresponding $P_{context}$:

1. **Scene Caption:** `Given a script starting with "Scripts:", where a person describes objects in an indoor scene ...`

2. **Object Caption:** `Based on a script beginning with "Scripts:" detailing objects in an indoor scene ...`

3. **Object Entites:** `Using the "OBJECT LIST:", which describes objects and their bounding boxes in an indoor environment in the format: object [position x, position y, position z, length x, length y, length z] ...`

4. **Object bounding boxes:** `Provided is a list titled "OBJECT LIST:", which details objects and their bounding boxes in an indoor setting, formatted as: object [position x, position y, position z, length x, length y, length z] ...`

We generate 12 types of instruction following annotations. In order to reduce hallucination, we only use some types of contexts for each type of data generation. Here are the annotation types, the corresponding $P_{instruction}$ and the used context types.:

1. **Scene Caption:** `provide a summarization starting with "Summary:". The summary should detail the objects, their positions, appearances, and the function of the room. Scripts:` Used contexts: scene caption, object entities, object bounding boxes.

2. **Object Caption:** `provide a summarization starting with "Summary:". The summary should detail the objects, their positions, appearances, and the function of the room. Scripts:` Used contexts: object caption.

3. **General Question Answering:** `visualize a scenario with a Human and a robot assistant present. Construct a Question-Answer pair, where the Human asks and the robot answers. Format as follows: Q: [Human's question based on the script details] A: [Robot's answer based on the script].` Used contexts: scene caption, object caption, object entities, object bounding boxes.

4. **Question Answering(Concept):** `Produce question-answer pairs wherein the human inquires about the existence of the objects. Q: [Human's question about the existence of objects] A: [Robot's response based on the script details].` Used contexts: scene caption, object entities, object bounding boxes.

5. **Question Answering(Concept-Negation):** `Generate question-answer pairs where the human asks about objects that are not present in the given list. The answers should confirm the absence of these objects. Q: [Human's question about the existence of objects] A: [Robot's response based on the script details].` Used contexts: scene caption, object entities, object bounding boxes.

6. **Question Answering(Counting):** `Produce question-answer pairs wherein the human inquires about the number of the objects. Q: [Human's question about the number of objects] A: [Robot's response based on the script details].` Used contexts: object entities, object bounding boxes.

7. **Question Answering(Spatial):** `picture a setting with a Human and a robot assistant. Create question-answer pairs wherein the Human inquires about the location of specific objects. Frame your dialogue in this manner: Q: [Human's question about an object's position. A: [Robot's answer referring to the object list].` Used contexts: scene caption, object bounding boxes.

8. **Question Answering(Comparison):** `Produce question-answer pairs wherein the human inquires about differentiating aspects of the objects. The queries should compare the number of the objects./The queries should compare the size of the objects./The queries should compare the functionalities of the objects./The queries should emphasize the distinctions between the objects. Please structure the dialogue as follows: Q: [Human's question distinguishing the objects from the script] A: [Robot's response based on the script details].` Used contexts: scene caption, object captions, object entities, object bounding boxes.

9. **Question Answering(Navigation):** envisage a scenario where a robot navigates the area based on human instructions. Generate question-answer pairs that relate to navigating within the scene. The robot may inquire about directions, routes, and objects encountered along the way. Please adhere to the following format: Q: [Robot's question about navigation relative to the objects in the script] A: [Human's answer guiding the robot, using information from the script]. Used contexts: scene caption, object bounding boxes.

10. **Human-robot Dialogue:** envision a scenario with a Human and a robot assistant interacting. Create a multi-round dialogue between them. Begin the human's lines with "Human:" and the robot's lines with "Robot:"." Used contexts: scene caption, object entities, object bounding boxes.

11. **Task Decomposition:** suggest daily tasks or chores relevant to the described environment. Each task should include a concise description and a step-by-step guide on how to complete it using only objects mentioned in the scene. Answer using this format: Task: [Brief description of the task Step-by-step instruction: 1.[First step] 2.[Second step] 3.[Third step]. Used contexts: scene caption, object entities, object bounding boxes.

12. **Functionality Improvement:** suggest ways to enhance the functionality of the environment. Your recommendation should focus on improvements like better lighting for reading. Provide your answer in the given format: Function: [Aspect of the environment to enhance] Method: [Suggestion for improvement based on the described scene]. Used contexts: scene caption, object entities, object bounding boxes.

11.2. Data Post-processing

To process the instructional data generated from stage 2, we use LLama-2-chat-70b [10] to remove any redundant and inaccurate information. Here is the prompt we use at this stage: Correct errors in the provided paragraph. Eliminate repeated sentences, meaningless characters, and non-English phrases. Remove unnecessary repetition. Complete incomplete sentences. Provide the corrected answer directly without additional explanation.

11.3. Instruction-type Distribution

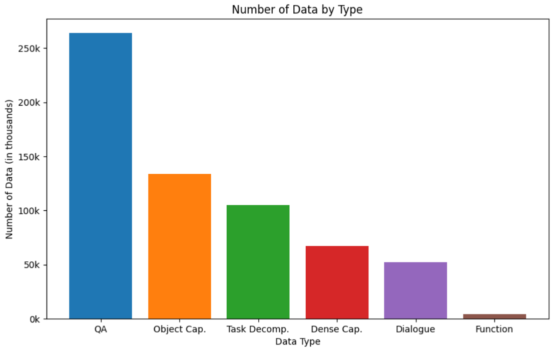


Figure 5. Distribution of the scene data by each type of instruction.

Fig. 5 shows the distribution of each type of instructional-following annotations, including question answering(QA), object caption, tasks decomposition, scene caption, human-robot dialogue and function improvement.

12. Training and Inference Detail

12.1. Projection Layer Structure.

The feature of each point is a concatenation of CLIP-H/14 feature, point color, and point position, which is of 1030 dimension. The input dimension to the LLM is 768 dimension. The projection layer is composed of a fully connected layer of a matrix shape $[1030, 768]$, followed by a GELU activation layer, and then is a fully connected layer of a matrix shape $[768, 768]$.

12.2. Pre-training Projection Layer.

We use $32\times$ NVIDIA A100 GPU to train the projection layer. The total batch size is 64 at this stage. We use AdamW as the optimizer, and a learning rate of $1e-5$ to train the projection layer. The training starts with 1000 steps of warmup with a warmup learning rate of $1e-6$. The training takes a total of 5 hours.

12.3. Finetuning the Projection Layer with the LLM.

We use $32\times$ NVIDIA A100 GPU to finetune the projection layer with the LLM. The total batch size is 64 at this stage. We use AdamW as the optimizer, and a learning rate of $2e-5$ to finetune the model. The training starts with 2000 steps of warmup with a warmup learning rate of $1e-6$. The fine-tuning takes a total of 8 hours.

12.4. Task-specific Finetuning.

We use the same setting for all benchmarks for task-specific tuning. We use $32\times$ NVIDIA A100 GPU to finetune the projection layer with the LLM. The total batch size is 64 at this stage. We use AdamW as the optimizer, and a learning rate of $2e-5$. For the tuning steps, we report the result of $1,500$ step for ScanQA and SQA3D, $3,500$ step for Alfred, $6,000$ step for Scan2Cap, and $5,000$ step for 3DMV-VQA.

12.5. Inference Time.

We use a batch size of 1 to mimic a real interactive setting. The inference speed is 23.7 tokens/second on an Nvidia RTX 3090Ti.

References

[1] AI2-THOR: An Interactive 3D Environment for Visual AI. *ArXiv*, abs/1712.05474, 2017. 5

[2] Daichi Azuma, Taiki Miyanishi, Shuhei Kurita, and Motoaki Kawanabe. Scanqa: 3d question answering for spatial scene understanding. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19129–19139, 2022. 2

[3] Jun Chen, Deyao Zhu, Xiaoqian Shen, Xiang Li, Zechun Liu, Pengchuan Zhang, Raghuraman Krishnamoorthi, Vikas Chandra, Yunyang Xiong, and Mohamed Elhoseiny. Minigpt-v2: large language model as a unified interface for vision-language multi-task learning. *arXiv preprint arXiv:2310.09478*, 2023. 5

[4] Zhenyu Chen, Ali Gholami, Matthias Nießner, and Angel X Chang. Scan2cap: Context-aware dense captioning in rgb-d scans. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3193–3203, 2021. 3, 4

[5] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Niessner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 3, 5

[6] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. *arXiv preprint arXiv:2307.12981*, 2023. 3, 4

[7] Xiaojian Ma, Silong Yong, Zilong Zheng, Qing Li, Yitao Liang, Song-Chun Zhu, and Siyuan Huang. Sqa3d: Situated question answering in 3d scenes. *arXiv preprint arXiv:2210.07474*, 2022. 2

[8] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*, 2016. 5

[9] Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. *arXiv preprint arXiv:2109.08238*, 2021. 4, 5

[10] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 2, 3, 7

[11] Wenhan Xiong, Anchit Gupta, Shubham Toshniwal, Yashar Mehdad, and Wen-tau Yih. Adapting pretrained text-to-text models for long text sequences. *arXiv preprint arXiv:2209.10052*, 2022. 2

[12] Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, Madian Khabsa, Han Fang, Yashar Mehdad, Sharan Narang, Kshitiz Malik, Angela Fan, Shruti Bhosale, Sergey Edunov, Mike Lewis, Sinong Wang, and Hao Ma. Effective long-context scaling of foundation models, 2023. 2