# LoreWeaver: A Novel Generation Multimodal LLM

Basel Anaya[1], Yazeed Mshayekh[1], Osama Awad[1], Ahmad Abu Diak[1], Subhi Abdullah[1], Abdelrahman Alkhatib[1] and Mohammed Ali[1]

[1] *University of Jordan, Department of Artificial Intelligence, Amman, Jordan*

## I. Introduction

In recent years, language models have made significant strides in natural language processing tasks such as text summarization, translation, and question-answering. However, generating coherent and engaging stories has remained an elusive challenge due to the complexity involved in capturing context, character development, plot progression, and emotional arcs within a narrative structure. To address this gap, we propose LoreWeaver, a novel generation multimodal language model designed to create immersive fictional narratives using advanced techniques from large language models (LLMs) like Mistral LLM [1].

LoreWeaver utilizes Mistral LLM's [1] powerful text generation capabilities while incorporating additional functionalities to provide users with more diverse and accessible experiences. Specifically, we introduce a text-to-audio feature tailored towards visually impaired individuals, enabling them to enjoy generated content through auditory means. This innovation enhances inclusivity by ensuring equal accessibility to creative written works, regardless of visual ability.

The primary goal of LoreWeaver is threefold: firstly, to explore how state-of-the-art LLMs can be harnessed to craft compelling and original fiction; secondly, to develop robust methods to translate these texts into high-quality spoken word renditions; and thirdly, to evaluate both qualitatively and quantitatively whether the resulting system meets user expectations and fosters engagement across different modalities.

While implementing LoreWeaver, several challenges emerged related to text analysis, syntactic consistency, semantic coherence, and prosody management during text-to-speech conversion. These hurdles necessitated innovative solutions involving complex linguistic analyses and sophisticated machine learning algorithms. Despite these difficulties, we believe that addressing these concerns will lead not only to improved performance but also pave the way for future advancements in generative AI systems capable of producing richer, more nuanced multimedia outputs.

This paper introduces the concept behind LoreWeaver, its technical foundation rooted in Mistral [1] LLM, and the unique features it offers. By outlining the motivation, objectives, methodology, and potential impact of this project, we aim to spark curiosity among researchers, practitioners, and enthusiasts alike about the possibilities afforded by next-generation, multimodal language models in literary creation and consumption.

Contact data: Basel Anaya, baselanaya@gmail.com

## II. Structure of the Paper

In this research paper we introduce a Large Language Model that's capable of writing Novels/Stories with the ability to provide Text-to-Speech (TTS) for the impaired people. We achieved this by fine-tuning a pretrained LLM called Mistral 7B and concatenate it with a pretrained TTS Model called speechT5. In the upcoming sections we further explain the history of the novels generation, What datasets we have fine-tuned the model on, The model's architecture and the limitations we faced while fine-tuning Mistral 7B. Subsequently we evaluated LoreWeaver LLM on nine distinct datasets: MMLU, HellaSwag, WinoG, PIQA, Arc-e, Arc-c, NQ, TriviaQA and HumanEval. Our fine-tuning methodology includes adjusting the architecture, learning rate schedules, optimization algorithms, and other hyperparameters.

## III. Related Work

### *a. Overview*

The exploration of automated story generation in the field of Artificial Intelligence has been a subject of research for over 50 years. The study of storytelling systems has witnessed substantial growth, with a notable increase in the development of such systems in recent years. The challenge in story generation lies, in part, in the lack of a well-defined task from an AI/computational perspective. Unlike other tasks that clearly specify inputs and expected output characteristics for developing the Algorithms , the generation of stories lacks that clarity. When humans create stories, the inputs influencing the process are often not transparent.

### *b. Early Works*

The earliest known storytelling system, according to historical records, is the Novel Writer system, crafted by Sheldon Klein and his team in 1973 [2]. This groundbreaking system had the remarkable ability to generate intriguing murder mystery stories, set in a weekend party atmosphere, in a matter of mere seconds. With just a brief description of the world and character traits as input, Novel Writer could produce comprehensive, 2100-word tales boasting rich detail

| Model | Modality | MMLU | HellaSwag | WinoG | Arc-c |
|---|---|---|---|---|---|
| LLaMA 2 7B | Pretrained | 44.4% | 77.1% | 69.5% | 43.2% |
| LLaMA 2 13B | Pretrained | 55.6% | 80.7% | 72.9% | 48.8% |
| Code-Llama 7B | Finetuned | 36.9% | 62.9% | 62.3% | 34.5% |
| Mistral 7B | Pretrained | 60.1% | 81.3% | 75.3% | 55.5% |
| LoreWeaver | Finetuned | **64.12**% | **83.92**% | **78.37**% | **59.89**% |

**TABLE 1:** BENCHMARK RESULTS FOR DIFFERENT MODELS.

and depth in under 19 seconds.

Also when we talk about story generation history, one of the first ever made models was James Meehan's Tale-Spin story generator. James Meehan is one of the first people to create a story generator back in 1977 [2]. Tale-Spin was designed to create interactive, text-based stories in a fantasy world. It employed a knowledge representation system and depended on character goals as triggers for action to maintain a model of the story world and used a set of rules to determine the progression of the narrative based on user input. It was not a sophisticated model, but it laid the groundwork for the newer models.

#### *1. Modern examples:*

Sassbook AI Writer [3]: it does a very good job at generating AI stories. The major feature of this generator is that it gives the ability to select the genre and style of the content beforehand such as the following genres. (Original, Classic, Humor, Sci-Fi, Romance, Thriller, Realism).

#### *2. Novel Studio:*

Offers a suite of tools to manage every aspect of a novel. From plot conception to character development, structuring outlines to auto-generating chapters, Novel Studio [4] is your AI-powered companion on your writing journey. It allows you to seamlessly manage the integral components of your novel - plot, characters, outlines, and chapters. Simplifying the intricate process of novel-writing it also supports refinement option to save your narrative at any stage of the writing process.

#### *3. Toolbaz:*

Is a tool that uses artificial intelligence to generate unique and original stories based on input provided by the user. The tool uses natural language processing (NLP) [5] to analyze the keywords and themes you input, it then uses this information to generate a list of potential story ideas based on your specific interests and preferences. Toolbaz [4] uses machine learning algorithms to analyze patterns and trends in the data it has processed, allowing it to continually improve and generate even more accurate and relevant ideas.

### *c. Our early plan*

When we first considered a story generator project, the first model that came to mind was NextGPT [6]. It is a state-of-the-art model for text generation that uses GPT [7] (Generative Pre-Training Transformer [8]) with many strengths and weaknesses. however, we realized that a Large Language model such as Mistral 7B (See more details about it in figure 1) would be a better fit for the project for the following reasons:

- Output quality: Mistral 7B [1] tends to produce more coherent and contextually relevant text than Next GPT, especially for long-form text generation tasks.

- Efficiency: Next GPT is generally faster and more efficient than Mistral 7B[1], thanks to its simpler architecture and smaller parameter size.

Also, Mistral 7B[1] surpasses NextGPT[6] in these benchmarks which helped with the decision to use Mistral 7B[1]:

- Text Classification: Mistral 7B [1] achieves an accuracy of 96.5

- Question Answering: Mistral 7B [1] achieves an F1 [9] score of 92.8

## IV. METHODOLOGY

Through meticulous adaptation, configuration, and fine-tuning of Mistral 7B LLM[1] on Atlas Storyteller Dataset, we aspired to achieve significant breakthroughs in Novels and Stories generation. With a keen awareness of prevailing limitations and challenges for example the perplexity of the Model, the stability and coherency of the generated stories, our ultimate goal was to build upon existing knowledge, refine established approaches, and establish new benchmarks for excellence in fine-tuning a cutting-edge LLM for Novels generation.

The following sections outline the specific components of our methodology, delving deeper into the intricacies of our experimentation process, the tools deployed, the datasets utilized, and the evaluation strategies pursued. By maintaining transparency and thorough documentation throughout this section, we hope to facilitate future research endeavors and foster collaborative progress in the realm of Novels Writing.

Vast.ai[10] was our workstation the we used to rent the GPUs used for fine-tuning the model, It is a website that initialize a Jupyter notebook session running on a virtual machine supports powerful GPUs, and other components that are required to run our code. In order to make the model's weights as light as possible, we used different optimization techniques, like LoRa[11] which is a Parameter Efficient Fine-Tuning (PEFT)[12] technique that focuses on updating only a subset of the model's parameters. It often involves freezing certain layers or parts of the model. This approach can result in faster fine-tuning with fewer computational resources. Notable methods within PEFT[12] include

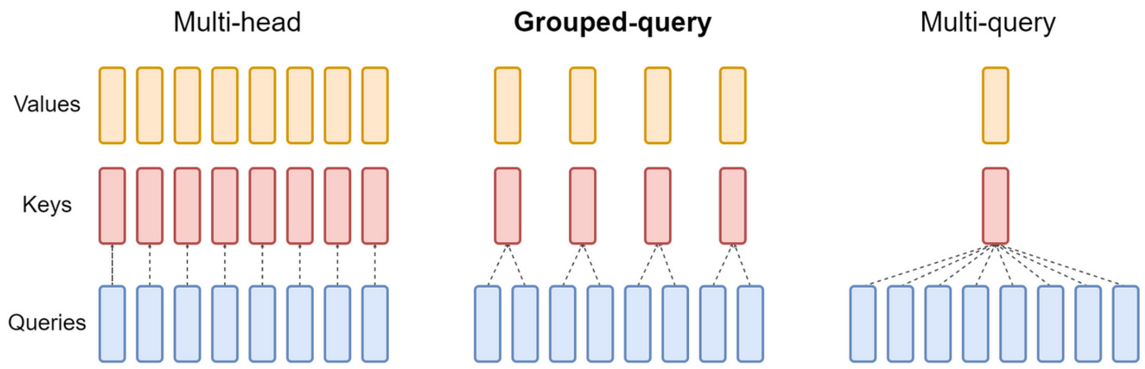


**Fig. 1:** Overview of grouped-query method. Multi-head attention has H query, key, and value heads. Multi-query attention shares single key and value heads across all query heads. Grouped-query attention instead shares single key and value heads for each group of query heads, interpolating between multi-head and multi-query attention.

QLoRa[13], LoRA, AdaLoRA[14], and Adaption Prompt (LLaMA[15] Adapter). It is suitable when the new task shares similarities with the original pre-training task. Recent state-of-the-art PEFT techniques achieve performance comparable to that of full fine-tuning[16].

Qlora[13] was used, which is an optimization method that allows for fine-tuning of large LLMs using just a single GPU while maintaining the high performance of a full 16-bit model in 4-bit quantization[17]. For instance, earlier, fine-tuning a 65 billion parameter model would require an equivalent of more than 780 gigabytes of VRAM. However, with QLoRA, fine-tuning a 65 billion parameter model requires just a single 48 gigabyte VRAM GPU, a significantly more attainable requirement.

SpeechT5[18] pretrained Text-To-Speech (TTS) model was our best selection among many other models and it stands out due to several reasons. Primarily, SpeechT5 combines the powerful capabilities of the T5 transformer architecture with a text-to-speech focus, delivering exceptional results. Key attributes that make SpeechT5 an outstanding option are:

- Versatility: Built on the highly flexible T5 transformer model, SpeechT5 accommodates various tasks efficiently. Its multi-task learning capability enables the model to handle text-to-speech conversion, speaker identification, and even phoneme duration prediction, ensuring comprehensive applicability.
- Superior Quality: Benefiting from the advanced T5 architecture, SpeechT5 consistently produces high-quality synthetic voices. It masters subtle variations, prosody, and emotion expression, granting users a life-like listening experience.
- Scalability: Owing to its transformer roots, SpeechT5 scales effortlessly with increasing dataset sizes. Thus, it capitalizes on larger text-to-speech corpora, continually boosting voice quality and reducing discrepancies.

First we load the data that we want to train the model on from huggingface website, then we apply auto_tokenizer from huggingface, we set pad_token to eos_token, then we train the model on A100 PCIE 80 GB VRAM GPU, it took like 1 hour to fine-tuned. The adapter, which is a small module or set of parameters that has been trained to adapt the pre-trained model to a specific task, was saved and uploaded to hugging-face, then we merged the adapter with the base model (Mistral 7B). Finally, we made an interface in order to generate stories as a text and/or a voice.

## *a. Model Architecture*

For our project we have used 3 main models the first model we have used is the grouped-query attention.

**Grouped-query attention (GQA)[19]**

As we can see from the figure above, it shows us 3 different configurations:

The first part is the Multi-head Attention[8], from its name it has multiple 'heads', where each consisting of its own keys, values, queries. This allows the model to have a deep insight of the Multi-head attention.

The second part is Group-query Attention[19] organizes query heads into distinct G groups, where each group shares a single key head and value head.

It's a model that solved 2 issues, first one is the traditional multi-head attention which has a great quality but a slow performance, second one is the multi-query attention where its performance is efficient and fast, but the quality is low.

For the last architecture it's the Multi-query Attention. This approach shares a single key and a single value head but does not capture the relationships as effectively as multi-head attention 1.

**Sliding Window Attention (SWA)[20]**

For our second model we have used is the Sliding Window Attention.

The first self-attention is the Full n2 attention which forms a complete matrix through taking every part of the input and attends to every other part which is computationally very expensive.

The second configuration is the Sliding Window Attention, where the main focus is restricted to a sliding window, which reduces the computation by giving all focus on a subset of a sequence at a given time.

The third configuration we have in the figure is the Dilated Sliding Window: This sliding window technique that skips elements to cover a wide range broader with fewer computations.

The fourth and last configuration which is the Global+sliding Window gives us a balance between comprehensive attention and computational efficiency by taking few globally important tokens **??**.

**Tranformer[8]**

For our third and last model we have used is the Transformers.

The above image demonstrates a Transformer model architecture[8], where its commonly used for tasks like language translation or text generation.

For the first stage, it's the Input Embedding of the raw input data such as: words, where its then converted into vectors that the model can interpret.

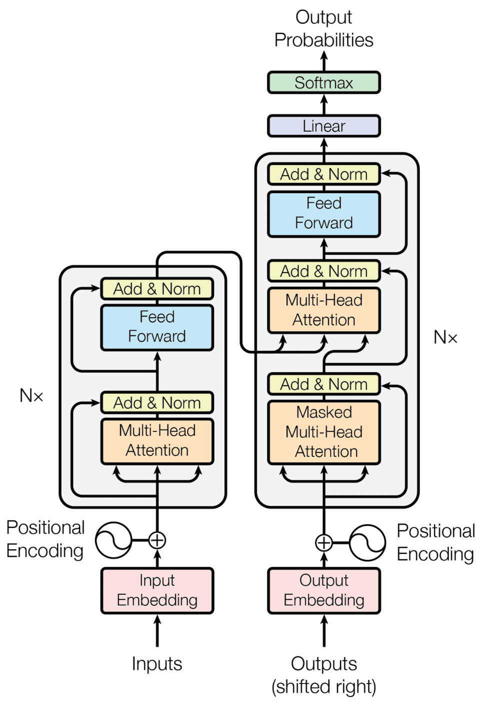


**Fig. 2:** Transformer Architecture

We then go to the positional encoding to process the data in sequence to know the order of the input.

There are N layers (N,x), each containing 3 parts:

Multi-Head Attention: This allows the model to pay attention to different parts of the input simultaneously. Feed Forward Networks: which takes the input to further processes after attention.

Add & Norm: Where they act as steps to help stabilize the learning process.

On the output side:

Output Embedding: We can think of it as input embedding, but for the output data.

Positional Encoding: It also adds ordered information to the output embeddings.

Multi-Head Attention: Prevents the model from "cheating" by looking at the answer.

| Parameters | Value |
|---|---|
| Activate 4-bit precision base model loading | **True** |
| LoRA attention dimension | **16** |
| Alpha parameter for LoRA scaling | **32** |
| Dropout probability for LoRA layers | **0.05** |
| Compute dtype for 4-bit base models | **torch.bfloat16** |
| Quantization type[21] (fp4 or nf4) | **"nf4"** |
| Activate nested quantization for 4-bit base models (double quantization) | **False** |
| Number of training epochs | **1** |
| Batch size per GPU for training | **4** |
| Batch size per GPU for evaluation | **4** |
| Number of update steps to accumulate the gradients for | **1** |
| Enable gradient checkpointing[22] | **True** |
| Maximum gradient normal (gradient clipping) | **0.3** |
| Initial learning rate (AdamW optimizer) | **2e-4** |
| Weight decay to apply to all layers except bias/LayerNorm weights | **0.001** |
| Learning rate schedule (constant a bit better than cosine) | **"constant"** |
| Number of training steps (overrides num_train_epochs) | **-1** |
| Ratio of steps for a linear warmup (from 0 to learning rate) | **0.03** |
| Save checkpoint every X updates steps | **25** |
| Log every X updates steps | **25** |

**TABLE 2:** ALL PARAMETERS THAT WE USED AND THEIR VALUES.

Finally, the output of the models goes through two functions: Linear Layer and a Softmax to turn the model internal representation into realistic predictions, such as the next word in a sentence 3.

## V. EXPERIMENTS

In Table 2, you can find the values of the parameters that we use.

In this section, we are going to talk about what are the models, dataset, hardware resoruce, time, and Out-Of-Memory/successful (you can see all details about experiments in Table 4).

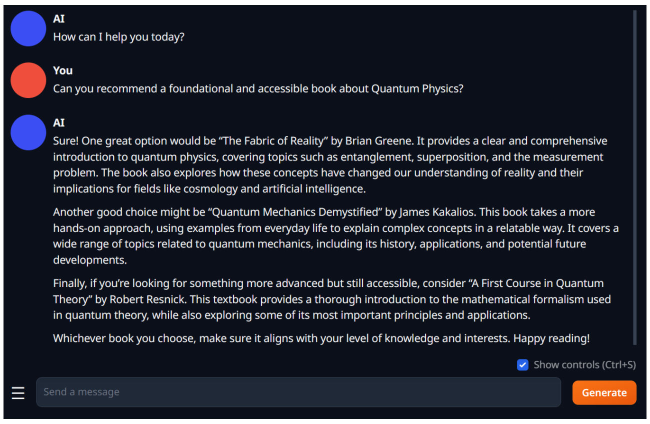
**Datasets**

To train our model, we used the "Atlas Storyteller" dataset[23] from Hugging Face, a wide and diverse collection with over 5,020 rows of narrative texts. Since we don't have that much supported resources to deal with large datasets we decided to use this dataset which we can deal with and train in our model. This dataset is specifically designed for advanced natural language understanding and generation tasks, offering a rich variety of stories set in different contexts. Each narrative is carefully crafted to provide a comprehensive background for language models to learn and adapt to diverse storytelling styles and linguistic intricacies, thus challenging and enhancing the capabilities of advanced language processing models.

Additionally, we employed the "Mistral-7B-v0.1" model, also from Hugging Face, known for its robustness and versatility in handling various language processing tasks, making it an ideal choice for our project. The integration of this specific dataset and model has enabled us to push the boundaries of automated storytelling, ensuring that our model not only understands the intricacies of language but also generates consistent, contextually rich narratives.

## VI. Results

We have tested LoreWeaver on several prompts as shown in Figure 3, And after the evaluating the model we came out with those metrics in Table 3



AI
How can I help you today?

You
Can you recommend a foundational and accessible book about Quantum Physics?

AI
Sure! One great option would be "The Fabric of Reality" by Brian Greene. It provides a clear and comprehensive introduction to quantum physics, covering topics such as entanglement, superposition, and the measurement problem. The book also explores how these concepts have changed our understanding of reality and their implications for fields like cosmology and artificial intelligence.

Another good choice might be "Quantum Mechanics Demystified" by James Kakalios. This book takes a more hands-on approach, using examples from everyday life to explain complex concepts in a relatable way. It covers a wide range of topics related to quantum mechanics, including its history, applications, and potential future developments.

Finally, if you're looking for something more advanced but still accessible, consider "A First Course in Quantum Theory" by Robert Resnick. This textbook provides a thorough introduction to the mathematical formalism used in quantum theory, while also exploring some of its most important principles and applications.

Whichever book you choose, make sure it aligns with your level of knowledge and interests. Happy reading!

**Fig. 3:** LoreWeaver prompt example

## VII. Limitations

While our model Mistral 7B model accommodates a context size up to 8,000 tokens, and comparing it with the "Yarn Mistral 7B 128k window size"[30] which has a context window of 128,000 tokens, this will cause limits in the output like incomplete responses, limitations in the storytelling, and not enough for writing novels. These constraints underscores the need for models with larger context windows.

The Mistral-7B model, while powerful, also presents significant challenges in terms of computational resources and financial costs. Operating such a large language model requires large computational power, often necessitating advanced GPUs. This hardware requirement leads to increased costs and we faced constraints with the high computational costs.

One significant limitation of our model is its exclusive focus on text-based processing, which lacks its applicability in multimodal tasks to generate images based on our story/novel prompt, Another limitation of our model is that it is trained on a relatively small dataset (5020 Row), which may restrict its ability to generate quite complex stories/novels.

## VIII. Conclusion

In summary, with LoreWeaver, a language model that marks a significant advancement in the field of narrative generation. Our dedicated efforts have focused on overcoming challenges in analyzing text and converting it into images represented story/Novel enabling the creation of captivating stories. It is worth noting that our inclusion of a text to audio feature promotes inclusivity by allowing impaired individuals to engage with the content we generate.

The paper provides an exploration of LoreWeaver's core framework, methodology and potential implications sparking interest in the development of next generation language models designed specifically for creation. By delving into the history of story generation spanning five decades we deliberately highlight Mistral 7B as a choice due to its impressive output quality and effectiveness compared to other contemporary models.

LoreWeaver represents a leap in AI driven storytelling advocating for inclusivity and accessibility. Moving forward our focus will be on refining LoreWeaver exploring avenues, for narrative generation and actively contributing to the advancement of advanced AI systems that craft engaging stories.

This research sets the foundation, for exploration, refinement and practical application in the field of narrative generation. It shows potential, for investigations, enhancements and real world use in creating inclusive AI models that excel in storytelling.

| Model | Modality | MMLU[24] | HellaSwag[25] | WinoG | PIQA[26] | Arc-e[27] | Arc-c[27] | NQ | TriviaQA |
|---|---|---|---|---|---|---|---|---|---|
| LLaMA 2 7B | Pretrained | 44.4% | 77.1% | 69.5% | 77.9% | 68.7% | 43.2% | 24.7% | 63.8 |
| LLaMA 2 13B | Pretrained | 55.6% | 80.7% | 72.9% | 80.8% | 75.2% | 48.8% | 29.0% | 69.6 |
| Code-LLama 7B | Finetuned | 36.9% | 62.9% | 62.3% | 72.8% | 59.4% | 34.5% | 11.0% | 34.9 |
| Mistral 7B | Finetuned | 60.1% | 81.3% | 75.3% | 83.0% | 80.0% | 55.5% | 28.8% | 69.9 |

**TABLE 3:** COMPARISON OF MISTRAL 7B WITH LLAMA. MISTRAL 7B OUTPERFORMS LLAMA 2 13B ON ALL METRICS, AND APPROACHES THE CODE PERFORMANCE OF CODE-LLAMA 7B WITHOUT SACRIFICING PERFORMANCE ON NON-CODE BENCHMARKS.

| Model | Dataset | Hardware Resource | Time | OOM/Successful |
|---|---|---|---|---|
| Yarn-Mistral-7b-128k | Bookcorpus | A100 PCIE 80 GB VRAM | - | OOM |
| Yarn-Mistral-7b-128k | Open-text-books | A100 PCIE 80 GB VRAM | - | OOM |
| Yarn-Mistral-7b-64k | Open-text-books | A100 PCIE 80 GB VRAM | 74 Hours | Unsuccessful |
| Mistral-7B-v0.1 | Open-text-books | A100 PCIE 80 GB VRAM | 60 Hours | Unsuccessful |
| Mistral-7B-v0.1 | Atlas-storyteller | A100 PCIE 80 GB VRAM | 1 Hour and 30 minutes | Successful |

**TABLE 4:** OUR EXPERIMENTS

## REFERENCES

[1] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier *et al.*, “Mistral 7b,” *arXiv preprint arXiv:2310.06825*, 2023.

[2] J. R. Meehan, “Tale-spin, an interactive program that writes stories.” in *Ijcai*, vol. 77, 1977, pp. 91–98.

[3] S. W. H. G. "AI Writer, Text Summarizer. "ai writer, text summarizer, story writer, headline generator". "https://sassbook.com/".

[4] "NovelStudio". "become a visual novels creator". "https://www.novelstudio.art/".

[5] J. O’Connor and I. McDermott, *"NLP"*. Thorsons, 2001.

[6] S. Wu, H. Fei, L. Qu, W. Ji, and T.-S. Chua, “Next-gpt: Any-to-any multimodal llm,” *arXiv preprint arXiv:2309.05519*, 2023.

[7] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, “Improving language understanding by generative pre-training,” 2018.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.

[9] Z. C. Lipton, C. Elkan, and B. Narayanaswamy, “Thresholding classifiers to maximize f1 score,” *stat*, vol. 1050, p. 14, 2014.

[10] V. AI. "rent gpus". "https://vast.ai/".

[11] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “Lora: Low-rank adaptation of large language models,” *arXiv preprint arXiv:2106.09685*, 2021.

[12] H. Liu, D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal, and C. A. Raffel, “Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 1950–1965, 2022.

[13] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, “Qlora: Efficient finetuning of quantized llms,” *arXiv preprint arXiv:2305.14314*, 2023.

[14] Q. Zhang, M. Chen, A. Bukharin, P. He, Y. Cheng, W. Chen, and T. Zhao, “Adaptive budget allocation for parameter-efficient fine-tuning,” *arXiv preprint arXiv:2303.10512*, 2023.

[15] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, “Llama: Open and efficient foundation language models,” *arXiv preprint arXiv:2302.13971*, 2023.

[16] K. Lv, Y. Yang, T. Liu, Q. Gao, Q. Guo, and X. Qiu, “Full parameter fine-tuning for large language models with limited resources,” *arXiv preprint arXiv:2306.09782*, 2023.

[17] R. Banner, Y. Nahshan, and D. Soudry, “Post training 4-bit quantization of convolutional networks for rapid-deployment,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[18] J. Ao, R. Wang, L. Zhou, C. Wang, S. Ren, Y. Wu, S. Liu, T. Ko, Q. Li, Y. Zhang, Z. Wei, Y. Qian, J. Li, and F. Wei, “Speecht5: Unified-modal encoder-decoder pre-training for spoken language processing,” 2021.

[19] J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebrón, and S. Sanghai, “Gqa: Training generalized multi-query transformer models from multi-head checkpoints,” *arXiv preprint arXiv:2305.13245*, 2023.

[20] I. Beltagy, M. E. Peters, and A. Cohan, “Longformer: The long-document transformer,” *arXiv preprint arXiv:2004.05150*, 2020.

[21] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, “Quantization and training of neural networks for efficient integer-arithmetic-only inference,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2704–2713.

[22] A. Andersson, N. Koriakina, N. Sladoje, and J. Lindblad, “End-to-end multiple instance learning with gradient accumulation,” in *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 2022, pp. 2742–2746.

[23] A. storyteller · Datasets at Hugging Face. "atlasunified/atlas-storyteller". "https://huggingface.co/datasets/AtlasUnified/atlas-storyteller".

[24] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, “Measuring massive multitask language understanding,” *arXiv preprint arXiv:2009.03300*, 2020.

[25] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, “Hellaswag: Can a machine really finish your sentence?” *arXiv preprint arXiv:1905.07830*, 2019.

[26] Y. Bisk, R. Zellers, J. Gao, Y. Choi *et al.*, “Piqa: Reasoning about physical commonsense in natural language,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 05, 2020, pp. 7432–7439.

[27] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, “Think you have solved question answering? try arc, the ai2 reasoning challenge,” *ArXiv*, vol. abs/1803.05457, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:3922816

[28] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer, “Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension,” *arXiv preprint arXiv:1705.03551*, 2017.

[29] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman *et al.*, “Evaluating large language models trained on code,” *arXiv preprint arXiv:2107.03374*, 2021.

[30] B. Peng, J. Quesnelle, H. Fan, and E. Shippole, “Yarn: Efficient context window extension of large language models,” *arXiv preprint arXiv:2309.00071*, 2023.