L-LLM: Large Language LEGO Models

Stanford CS224N CustomProject

Alex Wang
Department of Computer Science
Stanford University
alexjw3@stanford.edu

Calvin Laughlin
Department of Computer Science
Stanford University
calvin3@stanford.edu

Abstract

Our project introduces a multifaceted approach to generating novel LEGO instruction manuals in a text-based format. We leverage the vision capabilities of GPT-4o and fine-tune models such as GPT-3.5-turbo, Llama-2-7B-chat-hf, and Mistral-7B using a corpus of 90 existing text-based LEGO manuals. We detail our methodology, which includes fine-tuning these models on both existing and synthetically generated manuals from GPT-4o vision prompt engineering. Our contributions include a novel vision-to-text agent and the generation of new, small-scale LEGO instructions.

Using our custom dataset comprised of instructions, most human-created for Bricks for the Blind and some translated from PDFs, we finetune our models to generate instructions for simple LEGO builds such as cars, castles, houses, boats, and spaceships. Additionally, we parse through visual instruction sets native to the LEGO website and translate them into the text-based format, enhancing the Bricks for the Blind dataset with synthetic data.

For evaluation, we use two grading rubrics to score each generated build and instruction manual out of 100. GPT-4o evaluates the quality of instructions, while human scoring assesses the actual builds. We aim to highlight the creative potential of LLMs and their limitations in planning, creativity, and instruction. Results show that GPT-4o and fine-tuned LLama-2-7B have shown the most promise in novel instruction generation, but there is still much work do be done in planning and data gathering.

1 Key Information to include

- TA mentor: **Ryan Li**
- No external collaborators, external mentor, and not sharing project
- **Contributions** Alex Wang worked on the Mistral-7B fine-tuning, the vision-to-text pipeline, the GPT-4o prompt engineering, and the bulk of the evaluation. Calvin Laughlin worked on the GPT-3.5-Turbo fine-tuning, the Llama-2-7B fine-tuning, and the creation of our database. Both worked on the paper and the coordination.

2 Introduction

Our project of creating novel text-based LEGO instructions for the blind represents a significant challenge falling in the realms of accessibility, natural language processing (NLP), computer vision, and, most importantly, creativity. LEGO building instructions are inherently visual and rely primarily on images to guide builders through the assembly of the LEGO kit. This poses a substantial barrier for those with visual impairments, being largely unable to utilize the visual guides. Our project aims to bridge this accessibility gap by both converting visual LEGO instructions into detailed text-based formats and, our primary focus, generating novel text-based LEGO instructions utilizing the original corpus of text-based instructions and our synthetic visual-to-text instructions. BrevDev (2023)

The main difficulties of this task lie within the necessity to accurately describe complex visual information in a comprehensive and easy-to-understand manner. Visual LEGO instructions are about identifying pieces and their placements, understanding spatial relationships, colors, shapes, and the sequential steps required to build the LEGO sets. This requires sophisticated vision-to-text conversion techniques, an understanding of LEGO building processes, the needs of visually impaired users, and specific fine-tuning for instruction-oriented novel generation.

Within the current literature, we are yet to see any real novel LEGO generation attempts utilizing artificial intelligence, and we see an opportunity to advance the LEGO and overall creative experience. The main novelty in our approach lies in the ability to democratize creativity, allowing anyone to explore new LEGO designs and access the overall building experience. This not only expands the accessibility of LEGO building but also enriches the LEGO community with fresh, imaginative builds that can be shared and enjoyed by all.

3 Related Work

A paper that most influenced our work was "Balancing Specialized and General Skills in LLMs: The Impact of Modern Tuning and Data Strategy" by Zhang (2023). This paper highlighted the methodology for fine-tuning and evaluating large language models, which is the main technique we employed in our project. Some notable inspiration we took from this project were the evaluation techniques presented, such as human evaluation of outputs (presented as the gold standard) and some automated metrics, in our case GPT-4o. This paper also presents evaluation criteria, with the most relevant to our project being "Clarity", "Completeness", and "Concreteness", which was edited to "Detail" in our rubric.

4 Approach

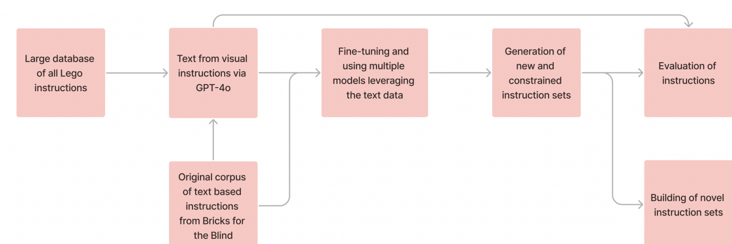

Figure 1: Workflow

Initially, we naively believed that we could create our own LLM from our dataset, but we did not have nearly enough examples nor the compute power to train our own LLM. So, we were initially recommended finetuning nanoGPT Karpathy (2023), a faster, smaller version of minGPT also trained on Wikipedia. After further research though, we found other models that were more advanced and better suited for our complex task of instruction generation.
We landed upon a few of the leading models featured on the LMSYS (2024) Chatbot Arena Leaderboard, including Mistral-7B, ranked 86th; GPT-turbo-1106, ranked 60th; LLaMA-7B-chat, ranked 73rd; our baseline GPT-4o, ranked 1st; and GPT-4o vision for the PDF image instruction to text translation. We chose these models because of their smaller size with most having only 7B parameters, making training more feasible with less compute, and because of their enhanced abilities shown through their leaderboard rankings, competing with much larger models, such as LLaMA-2-70B, ranked 50th. By finetuning atop these impressive models, we hoped to retain their learned weights and hopefully improve their areas of LEGO building by utilizing LoRA to only adjust relevant weights.
In addition, a sub-goal of our project was to create a pipeline that translated the PDF instructions created by LEGO into text-based instructions. We detail this method in **Figure 1**, and we hoped that by combining the already in-text examples with our synthetic examples from vision to create a solid-sized dataset for fine-tuning.

Mistral-7B Using the Hugging Face Transformer library and two H100 NVIDIA GPUs, we fine-tuned the Mistral-7B model with 4-bit quantization and Fully Sharded Data Parallel training. We tokenized our input text using AutoTokenizer which focused more specifically on the step-by-step instruction format and employed LoRA.

GPT-3.5-turbo-1106 We fine-tuned GPT-3.5-turbo-1106 using the OpenAI platform. Since the finetuning is somewhat a black box on their website, we only had control over a few of the hyperparameters including the number of epochs, LR multiplier, and batch size.

Llama-2-7B We fine-tuned LLaMA-2-7B, Meta's lightest LLaMA 2 version. Similar to Mistral-7B, we used the Hugging Face Transformer library and one T4 High-Ram GPU, with 4-bit quantization and Fully Sharded Data Parallel training. Again, we tokenized our input text using AutoTokenizer, and also employed LoRA.

GPT-4o We utilized GPT-4o through prompt engineering strategies such as chain of thought, task decomposition, and reflection to test the most state-of-the-art model and set a pseudo baseline for our other model generations.

GPT-4o Vision Translation Within our workflow of translating visual instructions, we utilized GPT-4o's vision capabilities to comb through a multitude of PNGs of LEGO instructions. We initially designate the PNG into 3 categories: `brick dictionary`, which provides every piece in the set; `instruction page`, the large bulk of the manuals; or `other`, non-important information within the LEGO manuals which aren't included. Based on the page's designation, we utilize GPT-4o again to translate the visual instructions into text-based instructions that can be utilized by the blind community. We emphasize the detailing of what bricks are used in each step and where they should be placed within the context of other bricks.

5 Experiments

5.1 Data

We use human-created LEGO text instructions, originally created for the blind and visually impaired, giving us just 91 examples ranging from 87 to 16.4k characters per example.
To obtain the text-based instructions in a format that was learnable by our models, we went through several different iterations. To begin, we manually separated the text-based instructions into different categories:

- `introduction`: A description of the visual and structural appearance of the LEGO set as well as its backstory and characters.
- `terms`: Contains common abbreviations for LEGO pieces (i.e. "Stud: the bump on a LEGO brick").
- `sorting`: Instructions for the assistants of the visually impaired on how to best situate the bricks for the builder's convenience.
- `instructions`: Detailed instructions walking the reader through the build.
- `misc`: Contains copyrights, advertisements, and general tips on how to assist the visually impaired with their build.

After we had separated these instructions into different categories, we chose to discard any elements that were repetitive, superfluous, or did not aid in the creation of the LEGO set. We created a script to turn these into a learnable format and initially formatted our training data as:

```
{"role": "system", "content": "You are a helpful assistant providing
detailed LEGO building instructions. Here are some terms you should
be familiar with: [TERMS]"},
{"role": "user", "content": "Provide step-by-step instructions for a LEGO
build with this description:[INTRODUCTION]"},
{"role": "assistant", "content": [INSTRUCTIONS]}
```

Where `system` is the instruction that goes to the model, `user` is the user's example prompt to the model, and `assistant` is the model's response. We adapted this training data slightly to fit the Chat Markup Language (chatML) format as well, and the structure was essentially the same aside from adding special characters to demarcate sentence beginning and end (`<s>` and `</s>`) as well as different titles for the headers (`[INST]]` and `[/INST]`).
This data format initially seemed promising, but upon further inspection, we see that this produced

examples with information that was not relevant to the build, along with long, rambling prompt examples that the model could not glean embedding relationships from. As such, we adapted our data cleaning pipeline and added API calls to GPT-3.5-turbo to simplify both the introduction and the instructions to turn them into more concise examples without losing detail or information. This created more realistic, learnable prompts.
One example of our dataset in chatML format can be seen below, with 2,053 tokens and 7,419 characters:

```
{"text": "<s>[INST] Generate LEGO instructions for building a model of the
iconic Tower Bridge, complete with its iconic towers, a working drawbridge,
and a red double-decker bus. [/INST] Book 1 - Bag 1: Large bag 1 contains
instructions for building the bridge.  \n- Step 1: Place a 32x16 blue flat
piece on the table to represent the Thames. \n- Step 2: Attach a 6x1 piece
                                                      ...
Thank you for your patience and creativity! </s>"}
```

The dataset can be viewed at calvinlaughlin/legobuilds-training.

5.2 Evaluation method

Since the task of LEGO instruction generation is somewhat subjective and does not have a numeric value associated with it, we had to get creative with how we scored our models' outputs. As such, we created two rubrics for scoring: an instruction rubric (Appendix A: **Figure 11**), and a build rubric (Appendix A: **Figure 12**). The instruction rubric is passed to GPT-4o, and the model is tasked with automatically generating a score based on our defined criteria. For each category, we pass GPT-4o three distinct examples with slightly tweaked outputs to get a range of creation. As an example, for the category "Car", we generate a "simple car," an "off-road automobile," and a "vintage car" with each of our models and report the average instruction score, defined as $\frac{s_{m_1}+s_{m_2}+s_{m_3}}{3}$ where s is each score of a model m_i, to test the breadth of the model to ensure it has not overfit to a certain seen build.
As for the output of the instructions themselves, i.e. the LEGO builds, we use a similar methodology of passing in the build rubric and the LEGO build to GPT-4o to generate a score based on the criteria. However, due to the human aspect of the creative and aesthetic criteria that we care about evaluating, we asked 15 humans for their evaluation of each build and averaged them for an overall score. For the LEGO builds themselves, we ran an evaluation only on the standard car build to have a standardized evaluation.
For our vision-to-text pipeline, we utilize a similar methodology of creating evaluation criteria and having it scored by GPT-4o, while also averaging it with our human evaluation of the instructions in comparison to their visual counterparts.

5.3 Experimental details

GPT-3.5-turbo-1106 To fine-tune GPT-3.5-turbo-1106, we used the OpenAI platform and formatted our data into the required format that matches their Chat Completions API. We ran 2 separate finetune jobs with the `auto` hyperparameter selection, which in both finetune jobs defaulted to 3 epochs, batch size 1, and an LR multiplier of 2. Our first finetune job, `legobuilder`, took in the original, long dataset containing 2,205,792 tokens. After observing that this model frequently outputs nonsensical instructions and repetitions, we cleaned our dataset with GPT and input a new, more controlled dataset containing 311,382 tokens. It's training loss can be seen in Appendix B: **Figure 20** This model performed significantly better and is the model we represent in our evaluation table. Its performance is still poor, though, and the finetuning seemed to make GPT-3.5-turbo perform worse than its non-finetuned counterpart. We suspect this is due to our limited dataset, and perhaps our model is overfitting to its seen examples. However, as seen in the builds, it can generate novel, interesting objects, unlike some of its other competing models.

Mistral-7B To fine-tune Mistral-7B, we used our custom training dataset as discussed previously with the Hugging Face Transformers library and 4-bit quantization. We conducted the training on 2 H100 GPUs with gradient checkpointing and k-bit training. Our learning rate was 2.5×10^{-6}, and we trained for 500 steps. We logged with `Weights and Biases`, as seen in Appendix B: **Figure 13**. We additionally employed LoRA, which allowed for fine-tuning on a smaller subset of parameters, thus significantly decreasing our computational and memory requirements. Additionally, focusing on more specific modules within the model, LoRA allows us to create an effective solution for resource-constrained environments. We tested over 15 different training configurations as well as token lengths, some results of which are found in the appendix.

LLaMA-2-7B-chat-hf To finetune LLaMA-2-7B-chat-hf, we used Google Colab running an L4 GPU with 22.5 GB of RAM. We adapted code from a LLaMA finetuning notebook created by Labonne (2024) to fit our needs and passed in our chatML formatted dataset. We tried a variety of different hyperparameter combinations, tweaking the number of epochs, the per-device train batch size, the gradient accumulation steps, the learning rate, and the maximum sequence length. Its final loss can be seen in Appendix E: **Figure 21**. We found through experimentation that when the number of epochs was too high (in our case 3), or the per-device train batch size was too low (in our case 1), the model outputs exhibited more repetition and nonsensical outputs than those with fewer epochs and a higher per device train batch size. We suspect this may be due to our very small finetuning dataset size. This would explain why more iterations caused the model to perform even worse since it is training on such a small subset of LEGO relationships that are widely non-generalizable.

Our codebase can be viewed at calvinlaughlin/LargeLanguageLegoModels

5.4 Results

Model	Structural Integrity (25 points)	Design Aesthetics (25 points)	Functionality (20 points)	Creativity (15 points)	Completeness (15 points)	Total (100 points)
Mistral-7B finetuned	23.1	23.1	17.6	10.8	12.9	87.5
Mistral-7B not finetuned	16.2	12.7	12.6	7.3	9.3	58.1
GPT-3.5 finetuned	17.3	7.1	5.6	9.1	4.4	43.4
GPT-3.5 not finetune	17.9	14.4	4.1	12.1	3.5	52.0
Llama-2 finetuned	20.9	20.5	18.0	13.8	13.9	87.1
Llama-2 not finetuned	21.3	18.3	18.6	13.8	11.4	83.4
GPT-4o baseline	21.1	18.5	17.3	13.2	7.7	77.8

Figure 2: LEGO Car Build Evaluation Scores

	Mistral-7B finetuned	Mistral-7B not finetuned	GPT-3.5 finetuned	GPT-3.5 not finetune	Llama-2 finetuned	Llama-2 not finetuned	GPT-4o baseline
Car average	65.33	59.00	36.75	80.33	55.67	86.67	80.33
Space ship average	63.67	73.00	37.00	82.16	36.0	85.33	83.33
House average	54.00	49.67	64.5	80.46	27.33	93.66	85.67
Boat average	62.33	71.00	54.0	79.33	37.0	87.33	87.67
Castle average	73.67	54.67	66.67	77.5	42.67	91.33	80.33
Total average	63.8	61.47	51.78	79.96	39.73	88.86	84.13

Figure 3: LEGO Instruction Evaluation Scores

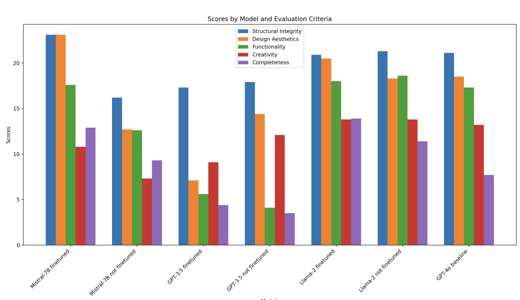

Figure 4: LEGO Car Build Evaluation Scores

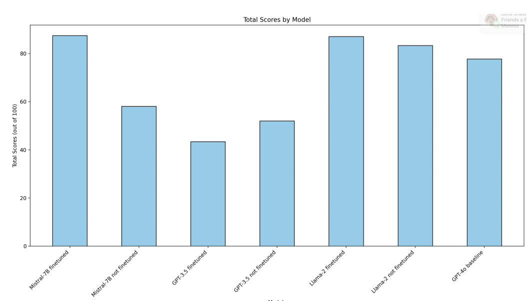

Figure 5: Total Scores of LEGO Car Builds

Generation Commentary The results of our LEGO instruction generation and the builds of those instructions are demonstrated in **Figures 2-9**. As shown in our build evaluation scores, our fine-tuned models outperformed their baseline counterparts, except for GPT-3.5-turbo. These results were as expected. However, in our instruction evaluation scores, the best results were shown by the GPT-3.5 not fine-tuned, Llama-2 not fine-tuned, and GPT-4o baseline models. These instruction results were worse than expected, as our fine-tuned models overall performed worse than their counterparts, except for the Mistral-7B fine-tuned model. Constructions of the LLM generated instructions can be viewed in **Appendix E**.

This discrepancy could be due to several factors, such as overfitting to our data, the overall quality and size of our dataset, the task being too specified, or an unnecessary increase in model complexity. Our approach likely should have focused more on the quality of data inputted to add novel and less

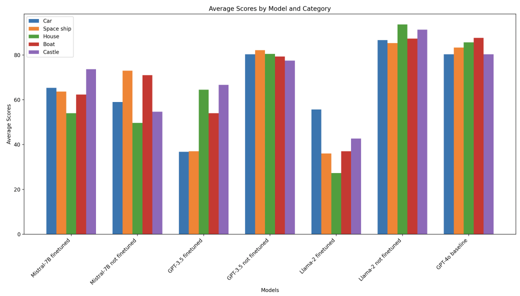

Figure 6: LEGO Instruction Evaluation Scores

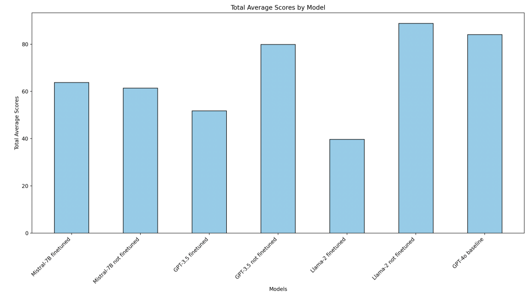

Figure 7: Total Scores of LEGO Instructions

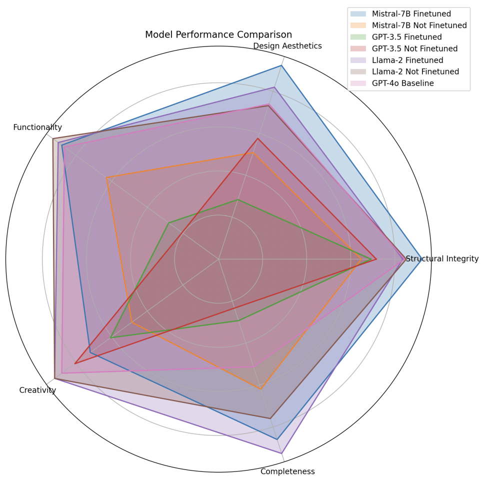

Figure 8: Build Radar Chart (scaled)

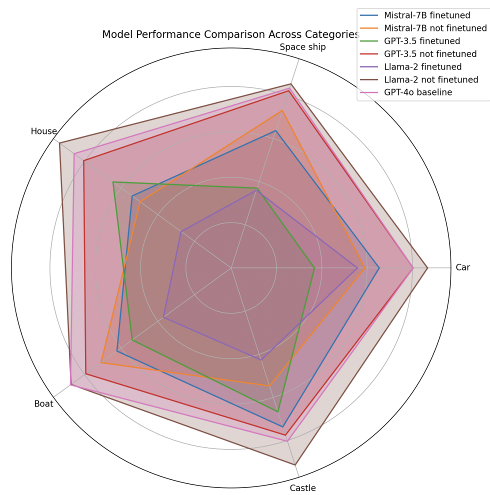

Figure 9: Instruction Radar Chart

noisy information into our baseline models, thereby increasing their effectiveness. However, due to the superior performance of our fine-tuned models in the actual resulting builds of our instructions, it is likely that our models were successfully fine-tuned to build better LEGO sets overall but fell short in generating clear instructions. The poorer performance in generating better instructions is probably due to overfitting.

Also, perhaps our use of GPT for evaluation possibly favors more typical outputs within common training datasets. In other words, GPT could have been favoring itself as well as other baseline, or common, models, especially since it too cannot perfectly generate LEGO instructions. This would be an interesting question to test but is outside the scope of this research.

Vision Commentary Our evaluation of the translated vision-based LEGO instructions to a similar text-based format as our original database is demonstrated in **Figure 10**. The evaluation showcases the effectiveness of our GPT-4o vision translations with the highest weight on the accuracy of the translations. This showcases GPT-4o's ability to translate visual instructions to text-based instructions in a relatively effective and accurate manner. This level of accuracy and efficiency surprised us and demonstrated GPT-4o's possibility of making real change and increasing societal efficiency.

6 Analysis

Creativity and Imagination Models like Llama-2-7B and Mistral-7B excelled in producing creative instructions that introduced new and unique LEGO builds. While some of our fine-tuned models were outperformed in their instruction evaluation, their performance in the builds they created demonstrated creative leaps. This creativity is crucial for maintaining the engagement and enjoyment of LEGO enthusiasts. However, we were unable to produce outputs that produced comprehensive builds that

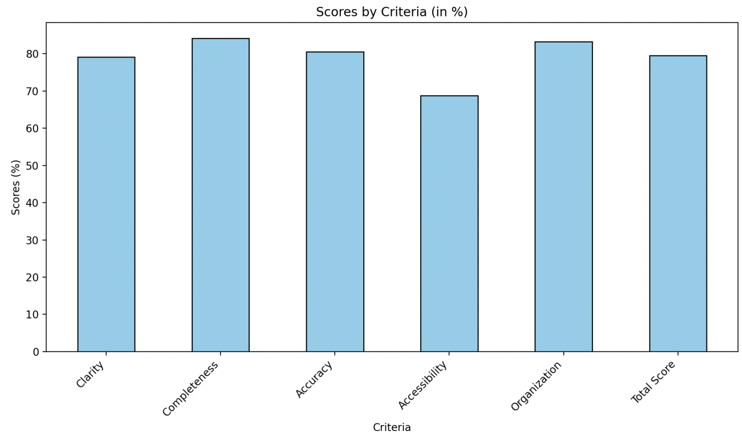

Figure 10: GPT-4o Vision Translation Results

we had not seen before, i.e. with detail and complexity that was higher than elementary school-level creativity.

Handling Complexity While our best models performed well on simpler builds, they struggled with generating more complex sets. Instructions for larger builds often included larger errors and lacked the needed detail for accurate assembly. They oftentimes lost sight of where they were in the process and where they needed to get to, repeating themselves or outputting nonsense. Overall, their ability to create a larger plan and then take the smaller steps to execute said plan was inadequate.

Efficiency of LoRA Fine-Tuning The use of LoRA fine-tuning proved effective in enhancing model performance without requiring extensive computational resources. This approach allowed for targeted improvements in instruction generation and reduced the cost of our training.

Piece Identification Errors All of our models misidentified LEGO pieces to some extent, providing incorrect LEGO piece codes while detailing piece requirements as well as within the instructions themselves. Further development in defining specific pieces is required and would likely allow for higher complexity builds as well.

Physical Placement Our models performed surprisingly well in the realm of understanding physical placement and location. Their detailing of where pieces needed to be placed in relation to cardinal directions and within the context of what had already been built was impressive in the smaller LEGO generations.

Creativity and Imagination Models like Llama-2-7B and Mistral-7B excelled in producing creative instructions that introduced new and unique LEGO builds. This creativity is crucial for maintaining the engagement and enjoyment of LEGO enthusiasts. However, we were unable to produce outputs that produced comprehensive builds that we had not seen before, i.e. with detail and complexity that was higher than elementary school-level creativity.

Evaluation Our evaluation methods were unique to our project and developed for our specific purposes. However, it was difficult for us to stray away from the gold standard of human evaluation, and felt that human evaluation was the most trustworthy causing us to validate all other outputs by our GPT-4o evaluations. We believe more work can be done in efficient, verified, and trustworthy evaluation methods.

7 Conclusion

Main Findings Our achievements include the fact that a majority of our fine-tuned models demonstrated superior performance in creating LEGO builds compared to their baselines. Additionally, Llama-2-7B and Mistral-7B excelled in generating creative and unique LEGO builds, maintaining the playful spirit of LEGO. The models showed a strong understanding of spatial awareness and physical placement of LEGO pieces as well. Furthermore, our utilization of LoRA proved effective and enhanced some of our model performance without requiring significant compute. Finally, GPT-4o successfully translated a large amount of visual LEGO instruction into a text-based, accessible format, showcasing promise for real-world applications for those who are visually impaired.

These achievements came with some challenges. Our fine-tuned models often overfitted, resulting in poorer performance in our text-based instruction evaluation from their baselines counterparts (besides Mistral-7B). Our limited and noisy dataset impacted our overall performance and generalizability of the model. Additionally, our models struggled with more complex LEGO builds and lacked the foresight to truly plan and execute on large ideas. Frequent misidentification of LEGO pieces plagued our instructions and highlighted our overall need for more refined training data and piece specification. Finally, we were somewhat reliant on human evaluation and verification for creative and aesthetic aspects of the project.

From this project, we learned that ensuring high-quality, diverse, and sufficient training data is crucial for improving model performance and generalization. Balancing model complexity with task-specific requirements is essential to avoid overfitting and maintain performance. Large language models (LLMs) have significant potential for creative tasks, but require careful tuning and evaluation to realize their full capabilities. Finally, and probably most importantly, we learned that robust and objective evaluation methods for creative tasks remain a challenge, but are critical for advancing AI applications in this domain.

Future Work We are confident that with more training data, the results would improve drastically. Presently, 90 text-based instructions do not contain enough information for the model to meaningfully create understandings and relationships. To create a model that truly assists the vision impaired with generating instructions will need a large magnitude more training and validation examples, which is very difficult given how varied and how few LEGO sets there are in the world.

8 Ethics Statement

Copyright An ethical challenge that we have come across is the potential infringement of intellectual property and copyright issues. This is an open debate in the generative AI world right now and is currently playing out in NYT v. OpenAI over claims of GPT being trained on NYT's intellectual property Harvard Law Review (2024). Generating this content that is too similar to copyrighted material could harm the brand and LEGO's commercial interests overall, especially if we provide alternatives to buying LEGO sets that we initially may have trained on.
To address these potential IP and copyright issues we could implement copyright detection filters that ensure the generated content doesn't replicate existing LEGO designs as well as emphasize original creations during the training process, thus encouraging unique outputs. Additionally, we could include overall legal disclaimers that our instructions are for personal and non-commercial usage.

Hallucinations Since the ultimate goal of these models is to generate novel, text-based instructions for the blind and visually impaired, the issue of hallucinations and falsified instructions could negatively impact the experience of a given builder. Since the experience of building LEGOs with limited vision is already difficult, it would add a whole new level of complexity if some of the instructions are incomplete, false, or contain pieces that do not exist.
As such, it is important that we mitigate hallucinations and false instructions as much as we can. Some ways we could do this would be by providing more fine-tuning data to the models, using larger base models, and adding negative examples to training for some reinforcement.

References

BrevDev. 2023. Mistral finetuning on own data. `https://github.com/brevdev/notebooks/blob/main/mistral-finetune-own-data.ipynb`.

Harvard Law Review. 2024. Nyt v. openai: The times's about-face. `https://harvardlawreview.org/blog/2024/04/nyt-v-openai-the-timess-about-face/`. Accessed: 2024-06.

Andrej Karpathy. 2023. nanogpt. `https://github.com/karpathy/nanoGPT`. Accessed: 2024-06.

Mathieu Labonne. 2024. Fine tune your own llama 2 model in a colab notebook. `https://mlabonne.github.io/blog/posts/Fine_Tune_Your_Own_Llama_2_Model_in_a_Colab_Notebook.html`. Accessed: 2024-06.

Calvin Laughlin. 2024a. Largelanguagelegomodels. https://github.com/calvinlaughlin/LargeLanguageLegoModels. Accessed: 2024-06.

Calvin Laughlin. 2024b. Lego builds training dataset. https://huggingface.co/datasets/calvinlaughlin/legobuilds-training. Accessed: 2024-06.

LMSYS. 2024. Chatbot arena leaderboard. https://huggingface.co/spaces/lmsys/chatbot-arena-leaderboard. Accessed: 2024-06.

et al Zhang, Zheng. 2023. Balancing specialized and general skills in llms: The impact of modern tuning and data strategy. arXiv preprint arXiv:2310.04945.

Appendix

A Rubrics

Criteria	21-25	16-20	11-15	6-10	0-5
Clarity (25 points)	Instructions are crystal clear, with precise descriptions of each step. Orientation and alignment of pieces are unambiguous, and language is simple and direct.	Instructions are generally clear but may occasionally lack detail in orientation or alignment. Most users can follow without significant confusion.	Instructions are somewhat clear, but several steps are vague or ambiguous. Users may need to re-read steps to understand.	Instructions are unclear, with frequent ambiguity in piece placement and orientation. Users may struggle to follow.	Instructions are very unclear, lacking essential details for understanding the build process.
Detail (25 points)	Highly detailed instructions with thorough descriptions for each piece and step. No assumptions are made about user knowledge.	Detailed instructions but some steps may lack minor details. Most necessary information is provided.	Instructions have a moderate level of detail. Some steps are not fully described, requiring user inference.	Instructions lack detail, with several steps missing important information. Users must often guess or infer next steps.	Instructions are missing many details, making it very difficult to follow the build process.
Organization (25 points)	Instructions are very well-organized. Steps are presented in a logical order with clear progression. Initial pieces list matches step-by-step instructions perfectly.	Instructions are well-organized but may have minor inconsistencies between the pieces list and step descriptions.	Instructions are moderately organized. There may be some jumps in logic or missing connections between steps.	Instructions are poorly organized. Steps are out of order or the progression is unclear, causing user confusion.	Instructions are very disorganized, with no logical flow. Users cannot follow the build process effectively.
Completeness (15 points)	N/A	N/A	The instruction set is complete, covering all necessary steps and pieces. No parts or steps are missing.	Instructions are incomplete, missing several steps or pieces. Users can still complete the build with some difficulty.	Instructions are very incomplete, with many missing steps or pieces. Users cannot complete the build without significant improvisation.
Accessibility (10 points)	N/A	N/A	N/A	Instructions are highly accessible. Includes helpful definitions, images, or diagrams. Suitable for all skill levels.	Instructions have limited accessibility. Beginners will struggle, and even intermediate users may face difficulties.

Figure 11: Instruction Rubric

Criteria	21-25	16-20	11-15	6-10	0-5
Structural Integrity (25 points)	The build is very stable and holds together well under normal handling. Pieces fit together perfectly without any gaps.	The build is stable with minor weaknesses. It holds together well but might have a few areas that are less secure.	The build is moderately stable. It has noticeable weak points that could cause it to fall apart with regular handling.	The build is unstable. Several parts are loose, and it may fall apart easily during normal use.	The build is very unstable and cannot hold together at all. Pieces fall apart with the slightest touch.
Design Aesthetics (25 points)	The design is visually appealing and creatively constructed. It demonstrates a high level of detail and craftsmanship.	The design is attractive with good use of colors and shapes. It shows thoughtful construction but may lack some intricate details.	The design is decent but may have some aesthetic flaws or lack creativity. Colors and shapes might not be as well-coordinated.	The design is somewhat unappealing. It lacks coherence in colors and shapes, and may appear sloppy or rushed.	The design is very unappealing. It shows no attention to aesthetics or creativity and looks poorly constructed.
Functionality (20 points)	N/A	The build functions exactly as intended. Any moving parts or interactive features work smoothly and reliably.	The build functions well with minor issues. Moving parts or interactive features mostly work but may have small glitches.	The build has significant functionality issues. Many moving parts or interactive features do not work properly.	The build does not function as intended. Moving parts or interactive features are broken or non-functional.
Creativity (15 points)	N/A	N/A	The build is highly creative, showing original and innovative use of pieces. It stands out as unique and imaginative.	The build is somewhat creative. It has a few unique elements but largely follows conventional designs.	The build shows no creativity. It is very basic and uninspired, following a standard or unoriginal design.
Completeness (15 points)	N/A	N/A	The build is completely finished with all intended parts and features included. No pieces are missing.	The build is moderately complete but missing several parts or features. The overall structure is still recognizable.	The build is very incomplete. Most parts or features are missing, making it unrecognizable and non-functional.

Figure 12: Build Rubric

B Appendix – Loss

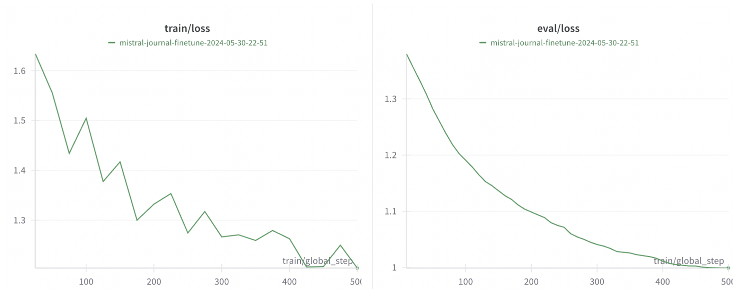

Figure 13: Mistral-7B Training Results

Step	Training Loss	Validation Loss
25	1.912500	5.681274
50	7.512000	8.630487
75	9.973000	10.820926
100	11.260400	11.494594
125	11.532900	11.537498
150	11.507500	11.454751
175	11.425100	11.397781
200	11.384100	11.364488
225	11.376800	11.341496
250	11.339900	11.322659

Figure 14: Mistral-7B Third Training

Step	Training Loss	Validation Loss
25	2.033100	1.934814
50	1.963300	1.934814
75	2.089600	1.934814
100	2.088200	1.934814
125	2.056800	1.934814
150	2.068500	1.934814
175	1.977800	1.934814
200	2.117700	1.934814
225	2.186000	1.934814
250	2.117800	1.934814
275	2.084700	1.934814

Figure 15: Mistral-7B Second Training

Step	Training Loss	Validation Loss
25	2.033100	1.934814
50	1.963300	1.934814
75	2.089600	1.934814
100	2.088200	1.934814
125	2.056800	1.934814
150	2.068500	1.934814
175	1.977800	1.934814
200	2.117700	1.934814
225	2.186000	1.934814
250	2.117800	1.934814
275	2.084700	1.934814
300	2.057800	1.934814
325	2.083500	1.934814
350	1.952400	1.934814
375	2.002200	1.934814
400	2.151900	1.934814
425	2.067300	1.934814
450	2.121000	1.934814
475	2.070200	1.934814
500	2.177700	1.934814

Figure 16: Mistral-7B First Training

Step	Training Loss	Validation Loss
10	No log	1.001915
20	No log	0.991909
30	1.235300	0.984333
40	1.235300	0.971782
50	1.212200	0.963916
60	1.212200	0.961445
70	1.212200	0.953186
80	1.122400	0.941355
90	1.122400	0.934100
100	1.231100	0.930023
110	1.231100	0.922518
120	1.231100	0.915865
130	1.109300	0.909297
140	1.109300	0.907883
150	1.172400	0.903107
160	1.172400	0.899314
170	1.172400	0.898016
180	1.070700	0.891575
190	1.070700	0.888703
200	1.105800	0.887078
210	1.105800	0.885939
220	1.105800	0.885530
230	1.127200	0.883819
240	1.127200	0.883687
250	1.076300	0.883126

Figure 17: Mistral-7B Fourth Training

Step	Training Loss	Validation Loss
10	No log	1.380144
20	No log	1.356094
30	1.633900	1.333002
40	1.633900	1.308959
50	1.555300	1.282141
60	1.555300	1.260235
70	1.555300	1.238446
80	1.434000	1.218665
90	1.434000	1.202601
100	1.504600	1.190747
110	1.504600	1.178900
120	1.504600	1.165126
130	1.377500	1.152990
140	1.377500	1.145584
150	1.417000	1.136560
160	1.417000	1.127714
170	1.417000	1.121009
180	1.299900	1.111175
190	1.299900	1.103743
200	1.332100	1.098715
210	1.332100	1.093714
220	1.332100	1.088854
230	1.353400	1.079697
240	1.353400	1.074852
250	1.274300	1.071528

Figure 18: Mistral-7B Fifth Training A

250	1.274300	1.071528
260	1.274300	1.060056
270	1.274300	1.054357
280	1.317300	1.050146
290	1.317300	1.044924
300	1.266400	1.040960
310	1.266400	1.038140
320	1.266400	1.034012
330	1.270500	1.028427
340	1.270500	1.027132
350	1.259200	1.025920
360	1.259200	Collapse Output [illegible]
370	1.259200	1.021119
380	1.279200	1.019380
390	1.279200	1.016576
400	1.262700	1.011455
410	1.262700	1.007411
420	1.262700	1.005263
430	1.206000	1.004064
440	1.206000	1.002777
450	1.206800	1.002646
460	1.206800	1.000799
470	1.206800	0.999746
480	1.249700	0.999395
490	1.249700	0.999026
500	1.203800	0.999146

Figure 19: Mistral-7B Fifth Training B

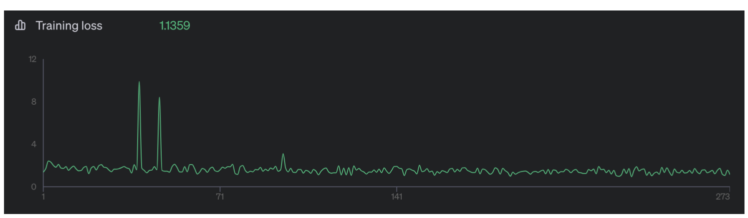

Figure 20: GPT-3.5-turbo-1106 Training Loss

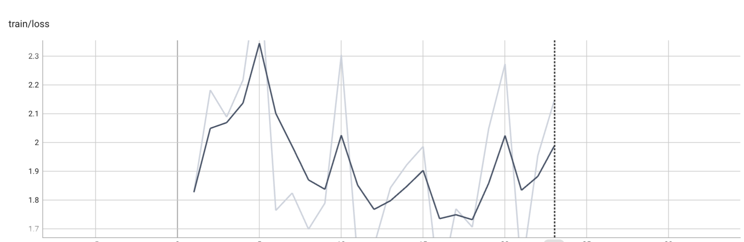

Figure 21: LLaMA-7B-chat-hf Training Loss

C Criteria for Evaluating LEGO Instruction Sets

Criteria	Points	Description
Clarity	15	Instructions are exceptionally clear, easy to understand, and free from ambiguity.
	12-14	Instructions are clear and mostly easy to understand with minor ambiguities.
	9-11	Instructions are generally clear but may require rereading to understand.
	5-8	Instructions are often unclear and confusing, making it hard to follow steps.
	1-4	Instructions are very unclear, with significant ambiguity and confusion.
	0	Instructions are incomprehensible or missing.
Completeness	15	Includes every detail necessary for assembly without referring back to the visual manual.
	12-14	Includes most details necessary for assembly, with only minor omissions.
	9-11	Includes many necessary details but has some noticeable gaps.
	5-8	Includes several necessary details, but major gaps make assembly difficult.
	1-4	Includes few necessary details, making assembly nearly impossible.
	0	Essential details are completely missing.
Accuracy	40	Accurately describes all parts, their placements, and assembly steps without errors.
	32-39	Accurately describes most parts and steps, with only minor errors.
	24-31	Describes parts and steps accurately but has some errors.
	16-23	Descriptions of parts and steps contain significant errors.
	8-15	Descriptions are mostly inaccurate, leading to incorrect assembly.
	0-7	Descriptions are completely inaccurate or missing.
Accessibility	15	Written in a way that is easily understood by individuals with no visual references.
	12-14	Mostly accessible, with only minor points that might need clarification for those with no visual aid.
	9-11	Moderately accessible, but several points may be confusing without visual references.
	5-8	Limited accessibility, with many points unclear without visual aid.
	1-4	Poorly accessible, making it very difficult to follow without visual references.
	0	Not accessible at all for individuals with visual impairments.
Organization	15	Logically organized with a clear, consistent structure that guides the reader through the steps.
	12-14	Well-organized, with only minor inconsistencies or jumps in the structure.
	9-11	Generally organized, but some parts may be out of order or structured inconsistently.
	5-8	Poorly organized, with many steps out of order or structured inconsistently.
	1-4	Very poorly organized, making it hard to follow the sequence of steps.
	0	No recognizable organization; steps are random or missing.

D Evaluation Prompts

All prompts follow this general structure:

```
Provide step-by-step instructions for a LEGO build with this description: [OBJECT].
Make sure that each step is doable in the real world and that each piece is a real
```

```
lego piece, include the part number.Firstly, state every lego piece that you will
need in the build. Then, create a small story about the build. Finally, create the
comprehensive step by step guide of the builds.
```

[OBJECT] refers to the desired end build, which we categorize into 5 groups, as seen below.

Car	Space Ship	House	Boat	Castle
a simple car	a simple spaceship	a simple house	a simple boat	a simple castle
a vintage car	a Star Wars style spacecraft	a cozy cottage	a sailboat	a medieval castle
an off-road automobile	a space shuttle	a countryside manor	a fancy yacht	a hilltop fortress

E LEGO LLM Build Visualizations

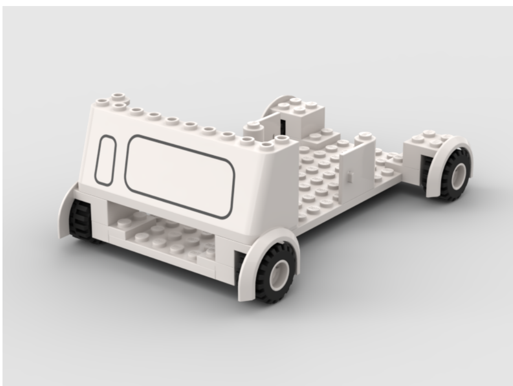

Figure 22: Mistral Baseline Car

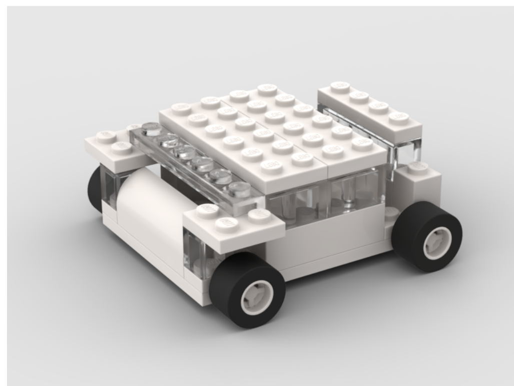

Figure 23: Mistral Fine-Tuned Car

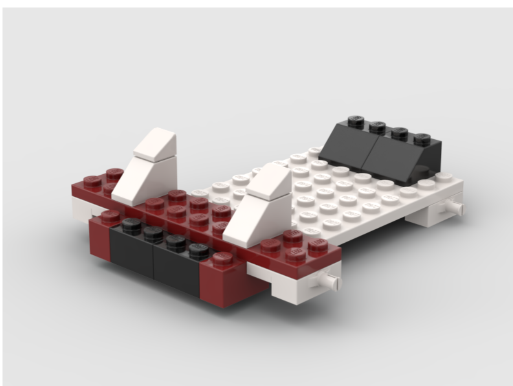

Figure 24: GPT3.5 Baseline Car

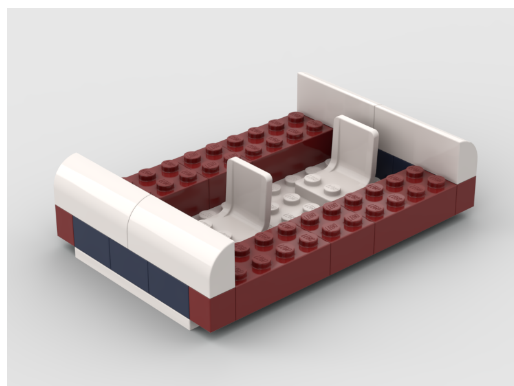

Figure 25: GPT3.5 Fine-Tuned Car

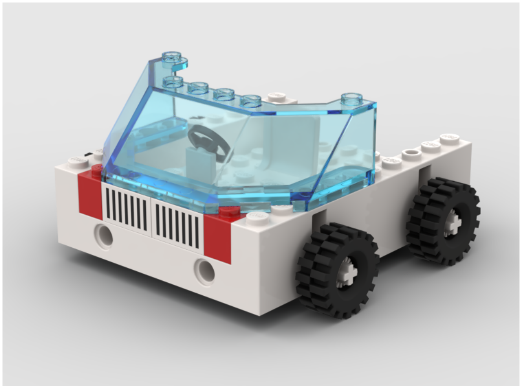

Figure 26: Llama2 Baseline Car

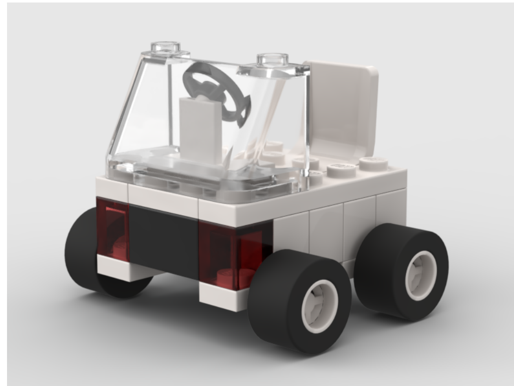

Figure 27: Llama2 Fine-tuned Car

Figure 28: GPT4o Car