Training LLMs to translate Mission Descriptions in Natural Language into Controller for Robot Swarms

Julian Jandeleit

November 18, 2024

Abstract

Translating global swarm objectives into individual controllers remains a significant challenge and can be called the micro-macro problem. Traditional approaches typically rely on manual modeling and domain-specific knowledge to create controllers that result in swarm behavior. Recent advancements in Large Language Models (LLMs) have shown promise in natural language processing, enabling these models to capture complex semantic relationships. Previous research used LLMs not only in natural language processing but also in the field of robotics, raising the question if the micro-macro problem can be approached using these models.
My thesis project already created a dataset of mission descriptions paired with simulator configurations to obtain corresponding controllers. The proposed master thesis aims to fine-tune a pretrained LLM on this dataset and evaluate its ability to generate swarm robot controllers from natural language mission descriptions.

Introduction

Collective robotics deals with a large number of robots working together on a common task. Typically, the robots are identical to each other and only have access to local information, without a global coordinator. This allows robots to serve as drop-in-replacements and robot swarms to have the potential to outperform single, more capable robots, in terms of efficiency by cooperation as well as in terms of robustness [Cheraghi et al., 2021].

Specifying a behavior for robots to execute on a swarm level, requires translating the global level to a local level, meaning the controller of each robot. Currently, there is no direct approach for this translation which can be called the micro-macro problem [Hamann, 2018]. Existing methods typically require a large amount of manual modeling, requiring time and domain knowledge. [Hamann and Wörn, 2008] statistically model individual robots including their peer-interactions and generalize to a swarm motion similar to statistical physics. Kuckling et al. [2018] compose elementary behaviors into a behavior tree using an evolutionary algorithm. The fitness function then has to be modeled such that the evolved behavior tree controller results in the desired global behavior.

Recently, the transformer model by Vaswani et al. [2017] has led to many advancements in the field of natural language processing. During training, language models build internal representations of the text that captures some of their semantic meaning [Mikolov et al., 2013]. This also works in transformer based models, capturing not only the meaning of single words but entire sentences [Devlin et al., 2019]. With enough parameters and trained on a sufficiently large corpus of text, they form so called Large Language Models (LLMs), that can solve many tasks without much explicit supervision [Radford et al., 2019]. This ability can be called emergent as it is typically not part in smaller versions of these models [Wei et al., 2022].

LLMs are now applied in many different fields including robotics. They can be used in tasks like perception, decision-making, control or human-robot-interaction [Zenga et al., 2023]. For example, Cao and Lee [2023] translate behavior trees from one domain to another and models robots capabilities as valid verbs in each step. Bucker et al. [2023] uses LLMs to capture user intent to change motion paths of robot arms. Here CLIP [Radford et al., 2021], a modified LLM that also incorporates images, is used as a link between text and the robots environment. Lykov and Tsetserukou [2023] finetunes a 7B model to translate natural language instructions into a behavior tree format. Yu et al. [2023] generate reward functions for Markov Decision Processes to bring robots into a pose specified by natural language.

This previous research raises the question if Large Language Models can capture the relationship between swarm behavior and robot controller to bridge the micro-macro gap. In contrast to previous works, they might have the ability to generate behaviors from one-shot prompt without the need for much manual modeling or domain expertise.
Achieving new behavior not included in the training set or from incomplete specifications given in natural language could indicate that the LLM would have not only memorized trained data but understood the relationship of how many interacting controllers form a global behavior.
To approach these questions in my thesis, I propose to finetune a pretrained LLM on mission descriptions and corresponding robot controllers and evaluate the results in comparison to ground truth and a dedicated test set of behaviors.

Methods

Based on the automatic design approach MAPLE by Kuckling et al. [2018], I created a dataset for mission descriptions and corresponding simulator configurations as my master project. The theoretical number is 80000 unique missions, including different types of variations from a few basis behaviors.

The dataset for swarm mission descriptions contains about 35% invalid configurations. These fall in mainly two cases:

- To short limit of generated output tokens
- Floating point numbers computed by the LLM to form a perfect circle out of walls were not exact enough so that robots can escape which crashes the simulator.

As more diversity in the dataset is preferable and behavior trees require even more exact handling of numeric values, some time should be spent to optimize the mission generator for numeric accuracy and find training methods to focus on these instead of the lexicographic configuration file similarity that was used before.

The configurations need to be turned into actual behavior trees using MAPLE. The dataset needs to properly be integrated into the simulator environment, potentially including the fitness functions for the following basis behaviors preset in the dataset:

- Aggregation
- Connection
- Foraging
- Distribution

The actually required work depends on the current state of the MAPLE implementation.

Then, the behavior trees and descriptions can be used to finetune a LLM as done in the master project with potential adjustments as discussed above. Splitting data into a train and test set enables to not only compare generated controllers with ground truth but to also estimate the generalisability of the finetuned model.

Finetuning LLMs needs ressources, as shown in the project report. They require GPUs with memory not provided on consumer products. These requirements can be loosened by e.g. quantisation methods and training adapters but still requires dedicated workstations with high performance graphics cards to work. In the project, I successfully used a NVIDIA Tesla P100 GPU with 16 GB VRAM. Generation of 224 mission configurations of max token length of 2500 took 8 hours. Availability of these resources naturally changes scope (duration of finetuning and number and size of experiments/comparison in the evaluation) of the project as well as timing of specific milestones.

Timeline

The working period of the thesis is from 15.08 to 01.03, where it is probably due (the final date is till to be determined by the central examination office). With the above mentioned constrains, the following is the targeted timeline for the thesis.

15.08.24

- Preparation of MAPLE for behavior tree generation with mission descriptions dataset
- Optimize dataset generator for handling numeric values (more time dedicated if previous step is complete early)
- Literature Research (related work, handling numeric values)

01.10.24

- (*?) MAPLE generation of behavior trees (monitoring and refinement)
- Preparation of LLM finetuning (data preprocessing, pipeline setup)
- First draft of methods section

01.11.24

- (*) Finetuning of LLM (monitoring and refinement)
- Extraction of initial results
- First draft of results section

01.12.24

- (*) Quantitative evaluation (Comparing LLM behavior trees with Maple behavior trees)
- (*?) Qualitative evaluation (Test generalization and missing data)
- Refine methods & results sections

01.01.25

- Final analysis and figures
- Writing of thesis (all sections)

01.02.25

- Proof reading of thesis & corrections
- Code cleanup & documentation

01.03.25

- Submission

Steps marked with a star (*) require high performance computation. Exact duration and points in time for training depend on provided resources and access timeslots. (*?) marked tasks probably work without dedicated hardware but some resources might be beneficial or necessary which is to be determined during setup phase as mentioned in methods section.

So some additional earlier access for setup reasons and estimation of scope would be helpful. Nevertheless, depending on progress and available resources, the entire finetuning process could start earlier to give more time flexibility to the evaluation. In case of gaps in access times, data preparation and inspection as well as part of the writing can be done in the meantime.

References

Ahmad Reza Cheraghi, Sahdia Shahzad, and Kalman Graffi. Past, present and future of swarm robotics, 2021. URL https://arxiv.org/pdf/2101.00671.pdf.

Heiko Hamann. *Swarm Robotics: A Formal Approach*. Springer, 2018.

Heiko Hamann and Heinz Wörn. A framework of space–time continuous models for algorithm design in swarm robotics. *Swarm Intelligence*, 2008. URL https://doi.org/10.1007/s11721-008-0015-3.

Jonas Kuckling, Antoine Ligot, Darko Bozhinoski, and Mauro Birattari. Behavior trees as a control architecture in the automatic modular design of robot swarms. In *Swarm Intelligence*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Neural Information Processing Systems*, 2017.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013. URL https://doi.org/10.48550/arXiv.1301.3781.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 2019. URL https://aclanthology.org/N19-1423.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, 2019. URL https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models, 2022. URL https://doi.org/10.48550/arXiv.2206.07682.

Fanlong Zenga, Wensheng Gana, Yongheng Wanga, Ning Liua, and Philip S. Yu. Large language models for robots: A survey, 2023. URL https://arxiv.org/pdf/2311.07226.

Yue Cao and C.S. George Lee. Robot behavior-tree-based task generation with large language models, 2023. URL https://doi.org/10.48550/arXiv.2302.12927.

Arthur Bucker, Luis Figueredo, Sami Haddadin, Ashish Kapoor, Shuang Ma, Sai Vemprala, and Rogerio Bonatti. Latte: Language trajectory transformer. 2023. URL https://doi.org/10.1109/ICRA48891.2023.10161068.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Artem Lykov and Dzmitry Tsetserukou. Llm-brain: Ai-driven fast generation of robot behaviour tree based on large language model, 2023. URL https://doi.org/10.48550/arXiv.2305.19352.

Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montserrat Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, and Fei Xia. Language to rewards for robotic skill synthesis. 2023. URL https://proceedings.mlr.press/v229/yu23a.html.