Automated Formalization of Biological Model Properties into Temporal Logics Using Large Language Models

Sumit Kumar Jha, Pranav Sinha and Sunny Raj

December 12, 2023

Automated Formalization of Biological Model Properties into Temporal Logics using Large Language Models

Sumit Kumar Jha
School of Computing and Information Science
Florida International University
Miami, FL, USA
jha@cs.fiu.edu

Pranav Sinha
Computer Science and Engineering
Oakland University
Rochester, USA
pranavsinha@oakland.edu

Sunny Raj
Computer Science and Engineering
Oakland University
Rochester, MI, USA
raj@oakland.edu

***Abstract*—In this paper, we demonstrate for the first time that large language models (LLMs) can be used to translate descriptions of biological model properties into formalized linear temporal specifications (LTL). We obtain these properties from published work on biological models and then use GPT-3.5 and 4 to formalize the description using LTL. Previous work decomposes the problem into multiple steps and utilizes multiple translation algorithms to perform the conversion. This decomposition of the translation task was needed with older neural networks and LLM models but is non-intuitive and can lead to compounding the accumulated errors. Our experimental evaluations show that state-of-the-art LLMs such as GPT-3.5 and 4 can successfully generate model specifications from descriptions without decomposing the translation into multiple sub-tasks and can provide an intuitive and convenient way to convert natural language into LTL specifications.**

***Index Terms*—Automated Formalization, LLM, AI**

I. Introduction

Linear temporal logic has been used to specify complex system behavior in multiple fields, including systems biology, robotics, and verification. Writing formal specifications of these systems is challenging, time-consuming, and error-prone even for experts in the field [1], and so there have been multiple attempts with varying degrees of success to automate the process of converting natural language descriptions into formal LTL specifications. While there is existing literature on natural language to LTS translation in fields like robotics and verification, work on biological models has been lacking and is the focus of this paper [1, 8, 6].

Early work on automated translation utilized machine learning techniques, while more recent work utilized transformers and large language models (LLMs) to perform the translations [1, 8]. However, most of these works decompose the translation into multiple steps, each step being realized using a specialized neural network or machine learning model. Most of these translations require some form of human feedback for proper functioning and still need expert involvement in the translation process. Furthermore, methods using LLMs require prompt engineering and few-shot learning as intermediate steps for the proper functioning of these translations. Powerful LLMs were not available during the publication of these papers, which could be a reason for their complexity. In this paper, we want to answer the simple question: can state-of-the-art LLMs automatically convert natural language into LTL specifications without needing constant expert human oversight? After performing experiments using biological descriptions in multiple published works, we answer the question in the affirmative, stating that LLMs can indeed generate these formalized LTL specifications.

II. Related Work

Some of the earliest work on automated LTL generation focused on converting structured English grammar into specification patterns and then into LTL [7]. Later attempts used SMT solving and semantic parsing [3]. State-of-the-art approaches have started utilizing neural networks and large language models to do the translation [1, 8]. However, all of these approaches are based on older models of GPT and thus require complex designs to perform translations. The nl2spec framework proposed by Cosler et al. uses LLM to decompose natural text into *sub-translations*, with each *sub-translations* having a confidence score that needs to be checked and edited by the framework user [1]. Once the *sub-translations* is verified by an expert, only then can the final translated LTL specification be generated. This approach requires prompt engineering for the LLM to perform the translation effectively.

The Lang2LTL framework proposed by Liu et al. similarly decomposes the translation into multiple sub-tasks, including named-entity recognition, grounding, and, finally, translation [8]. The named-entity recognition step identifies and replaces names with symbols, while the grounding task identifies environment propositions. The transformed text is then finally translated into LTL specification using the GPT-3 LLM. The method also provides a way of obtaining the accuracy of the conversions but does not seem to handle situations where the LLM generates a correct but differently worded LTL specification. We hypothesize that these sub-steps are unnecessary for a sufficiently powerful large language model. During our experiments with GPT-4, we observed that the named-entity recognition and grounding occur independently, and the model might produce multiple correct answers.

III. Background

A. Linear Temporal Logic

Linear Temporal Logic (LTL) is a formal logic that deals with the specification and verification of temporal properties in systems. It has operators that allow the precise expression of properties about the sequences of states in a system over time. LTL extend the propositional logic using the operators **N** (next) and **U** (until). LTL specifications can be written using the following grammar:

$$\phi ::= x \sim v \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \neg\phi \mid \mathbf{N}_{[a]}\phi \mid \phi_1 \mathbf{U}_{[a,b]} \phi_2$$

where $\sim \in \{\geq, \leq, =\}$, $v \in \mathbb{Q}$, and x is a state variable. **N** is the next temporal operator and **U** is the until temporal operator with time constraints $[a]$ and $[a, b]$. The formula $\mathbf{N}_{[a]}\phi$ holds if ϕ holds for the next a time steps. The formula $\phi_1 \mathbf{U}_{[a,b]} \phi_2$, holds if ϕ_1 holds until ϕ_2 holds at a future time instance. Other popular variants **G** (globally or always) and **F** (finally or eventually) can be constructed using **X** and **U**. $\mathbf{G}\phi$, specifies that ϕ must hold at all times, whereas $\mathbf{F}\phi$, specifies that ϕ must hold eventually, or at least once.

B. Large Language Models

Large language models, including GPT, BERT, T5, and Bloom, are built using the transformer neural network architecture and provide state-of-the-art performance on language-related tasks [11, 12, 2, 10]. LLMs are enormous in size, with GPT-3 containing around 175 billion parameters, GPT-3.5 containing more than double that, and GPT-4 estimated to have more than 1 trillion parameters. These models are often pre-trained on massive amounts of data, allowing them to learn intricate patterns and relationships that enable them to excel at language processing and translation. LLMs can act as repositories of information and have knowledge about wide-ranging subjects.

LLMs can perform extraordinary feats and then fail at seemingly simple tasks. Given the recent emergence and the apparent power of these LLM models, there is still a long way to go to understand their full capabilities and limitations. To remove some of the mystery, in this paper, we investigate if LLMs can be used for converting natural language descriptions of biological models into LTL specifications. We pick up these descriptions from various published works and then use LLM to generate the specifications. We then manually check the generated LTL specification for correctness, and we further contrast it with the LTL specifications in the published work. Through experimental evaluations, we find that LLMs generate multiple correct and some incorrect responses and are usually written using different LTL operators.

IV. Automated Formalization

We obtain biological model properties from multiple published documents and use both GPT-3.5 and GPT-4 to convert them into LTL specifications [9, 6, 4, 5]. To run GPT-4 we used the API from inside the python code. The prompt is as follows: "Convert the following text in Linear Temporal Logic, without using the Next operator: **Description**. Please type your answer in latex code." We require the output in latex as the LTL specification uses special symbols that need to be typed properly. GPT-3.5 refused to produce latex code using the API, so we used the ChatGPT interface to get the output. Examples of such conversions are listed below:

Description: Grb2 binds to FRS2 within 20 time units [6].
Published: $\mathbf{F}^{<20}(\mathrm{FRS2\_GRB} > 0)$
Response GPT-4: $\mathbf{F}^{<20}(\mathrm{Grb2BindsFRS2})$
Response GPT-3.5: $\mathbf{F}(\mathrm{T}\mathbf{U}_{\leq 20}(\phi))$

In the published work, $FRS2\_GRB > 0$ is understood as Grb2 binds to FRS2, GPT-4 being unaware of this, uses the symbol $\mathrm{Grb2BindsFRS2}$ to represent the same concept. Similarly GPT-3.5, make ϕ equal to Grb2 binds to FRS2. Both of the outputs listed above are correct, but the GPT-4 response is closer to the description in the published paper.

Description: G protein stays above the threshold of 6000 units for 2 time units and falls below 6000 before 20 time units [6].
Published: $\mathbf{G}^2(G > 6000) \wedge \mathbf{F}^{20}(G < 6000)$
Response GPT-4: $\mathbf{G}_{[0,2]}(G > 6000) \wedge \mathbf{F}_{<20}(G < 6000)$
Response GPT-3.5: $\phi\mathbf{U}_{\geq 2}(\neg\phi\mathbf{U}_{[0,20]}T)$

Here again, we see that GPT-4 produces results that are similar to the published work, while GPT-3.5 produces a response that looks correct at first glance but is actually incorrect. GPT-4 uses a subscript format to specify the time constraints compared to the paper, which uses a superscript. GPT-3.5 specifies ϕ to be equal to $G > 6000$, but its use of $\neg\phi$, which means $G \leq 6000$ in the equation instead of $G < 6000$, makes it incorrect. However, both of these models produce different responses for different runs, with GPT-4 producing the correct answer during most of the runs.

V. Conclusion and Future Work

We show that modern LLMs such as GPT-3.5 and GPT-4 can produce LTL specifications from natural language. Existing LLMs already understand LTL specifications, and we do not need to use few-shot learning to train them on it. We picked up descriptions of biological LTL specifications from published work and used GPT to translate them into formal LTL specifications. We observe that these models can convert natural language into LTL specifications, with GPT-4 performing better than GPT-3.5. However, we also observe variability in the response and see multiple versions of correct and some incorrect answers from the models. Future work along this line will focus in prompt engineering to decrease this variability while maintaining good translations. Any future work on testing the accuracy of these models will also have to consider the probabilistic behavior of these models.

REFERENCES

[1] Matthias Cosler et al. "nl2spec: Interactively Translating Unstructured Natural Language to Temporal Logics with Large Language Models". In: *arXiv preprint arXiv:2303.04864* (2023).

[2] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[3] Ivan Gavran, Eva Darulova, and Rupak Majumdar. "Interactive synthesis of temporal specifications from examples and natural language". In: *Proceedings of the ACM on Programming Languages* 4.OOPSLA (2020), pp. 1–26.

[4] Faraz Hussain et al. "Automated parameter estimation for biological models using Bayesian statistical model checking". In: *BMC bioinformatics* 16.17 (2015), pp. 1–14.

[5] Faraz Hussain et al. "EpiSpec: A formal specification language for parameterized agent-based models against epidemiological ground truth". In: *2014 IEEE 4th International Conference on Computational Advances in Bio and Medical Sciences (ICCABS)*. IEEE. 2014, pp. 1–6.

[6] Sumit K Jha et al. "A bayesian approach to model checking biological systems". In: *Computational Methods in Systems Biology: 7th International Conference, CMSB 2009, Bologna, Italy, August 31-September 1, 2009. Proceedings 7*. Springer. 2009, pp. 218–234.

[7] Sascha Konrad and Betty HC Cheng. "Real-time specification patterns". In: *Proceedings of the 27th international conference on Software engineering*. 2005, pp. 372–381.

[8] Jason Xinyu Liu et al. "Lang2ltl: Translating natural language commands to temporal specification with large language models". In: *Workshop on Language and Robotics at CoRL 2022*. 2022.

[9] "OpenAI. (2023). GPT3.5 [Large language model]." In: https://chat.openai.com.

[10] Colin Raffel et al. "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *The Journal of Machine Learning Research* 21.1 (2020), pp. 5485–5551.

[11] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[12] BigScience Workshop et al. "Bloom: A 176b-parameter open-access multilingual language model". In: *arXiv preprint arXiv:2211.05100* (2022).