Evaluating the Scalability and Reliability of a NARS-LLM Hybrid Natural Language Interface

Amitai Goldmeer & Neil Conley
Temple University
Philadelphia, PA

May 2, 2025

Abstract

Since the modern development of LLMs in the AI space, students and working professionals of all varieties have been turning to them for potential problem solving capabilities. However, LLMs are less 'problem solvers' and more 'text predictors'. When confronted with mistakes, or disagreeing evidence about claims that it has made, most LLMs are likely to immediately either revert on their decision, or double down with confidence, regardless of correctness. In this project, we seek to understand the greater differences between LLMs and AGI in how they handle logical inconsistencies. We hope to understand, and contribute, to the understanding of the limitations that come with both LLMs and AGI, and how this understanding can be used to make more robust and uncertainty-aware models.

We feel that this project is sufficiently different from other projects in this realm of study (ex: NARS-GPT) due to a difference of focus, and implementation. We aim to build a practical system to try and evaluate the scalability of NARS as a truth source to create a more reliable system, whereas NARS-GPT was done to prove that a system of this type could be created, and feasibly work. We are looking to 'retread' some of the work done by the original NARS-GPT team, to utilize LLMs that are notably more advanced than the ones that were used in the original project (such as local models of DeepSeek or Llama), especially since this technology has moved so quickly since the introduction of the project.

1 Problem Statement

Large Language Models (LLMs) have changed the AI space for students, researchers, and professionals. It has partially replaced the way that people search for information, gather and refine ideas, and solve problems. However, even with new models coming out frequently, it's accuracy can be called into question often. This is especially in areas where there is contradictory information, or

where it is forced to handle legitimately new ideas.

LLMs are far better to be considered text predictors and generators. While they are sophisticated in their own right, it can be hard to consider them legitimate reasoning engines. However, their natural language capabilities do offer insights and solutions for turning abstract reasoning engines into human readable responses. We believe then, that to combat the accuracy issues caused by LLMs, that the use of an actual reasoning engine such as NARS as an underlying layer for an LLM can greatly improve it's accuracy for users.

2 Technical Specifications

2.1 NARS

In order to create a NARS-LLM Hybrid Interface, there were a number of required systems. Firstly was a version of NARS that would meet our needs. Multiple options were considered, including NARS Python, and OpenNARS for Applications (ONA). NARS Python had some advantages in how it processed data and output the connection tables. Additionally, it's main language being Python would've made interfacing LLM's with it near trivial. However, for this project, ONA was chosen as the NARS model.

OpenNARS for Applications runs in a headless mode by default, and is very modular, allowing for easy modification. Additionally, it handles data manipulation very well, allowing both developers and users to understand how it is processing and understanding input data, as well as any data it returns as output. Running in a headless mode meant that we could create our own user interface without much trouble, which was a major advantage compared to NARS Python, which would've required developing a headless mode for it instead of using the built-in GUI.

2.2 LLM

There are a number of methods right now for the implementation of LLMs. Instead of using an API for an LLM such as ChatGPT or Gemini, we've instead opted to use one of the models provided by the Ollama service. Ollama allows for the free usage of localized LLMs. While it does create a slower system than using an API, which would allow near instantaneous responses, the usage of Ollama services allowed us to freely experiment with a number of LLM models in attempts to see what might work the best.

This project currently uses Deepseek-R1 as it's Large Language Model. It balances size so that any user can run it locally without too much slowdown or resource usage, while also maintaining high standards of accuracy. Other models, including numerous versions of Llama (Meta's LLM system), were experi-

mented with, however they gave subpar results. Working with better hardware would allow for larger models to be used which could potentially allow for better extraction of facts and results.

A secondary model is used particularly for **fact extraction**. When natural language statements are input, they need to have the underlying facts extracted so that they can be translated to Narsese. We found that attempting to use a model that is too large, or too general, can actually slow down the system and create worse results than using a smaller, more specialized model. Due to this, the secondary model utilized for fact extraction is Mistral AI, one of the models offered by Ollama.

2.3 Hybridization

The main goal of this project was to create a working hybridization between NARS as a reasoning system, and a LLM as a natural language interface. This was done by using NARS as an intermediary between the LLM on either side.

When running the project, a user can provide a statement, or ask a question, using natural language. The LLM model will then extract facts or questions from that natural language statement. These are then translated to Narsese, and feed it to NARS. We then allow for NARS to run for a certain amount of processing time, so that it can take the given inputs, and generate statements and conclusions between the new information, and it's current knowledge base.

The most confident outputs based on both truth and confidence values assigned by NARS are translated back into natural language statements, and returned to the LLM. Based on the output as context, and the initial user query, it takes the information and formulates a response to the user query. The use of the Ollama system lets us custom create system instructions, context, conversation history, and more. This allows far more control than an API call to ensure that the LLM does not hallucinate or give inaccurate results, and instead works faithfully with the NARS model for the best results.

3 Potential Limitations

While there are places where this approach to hybridizing NARS and LLMs can be very effective, it does have some notable limitations.

3.1 Reliance on Initial Information

One of the major advantages of NARS is that it requires effectively no initial information. It can work with a very small dataset of statements and facts, and work from there. However, when integrating with a LLM, the situation changes. Because we are using the LLM as a synthesizer and natural language translator,

it needs more context than NARS would alone. For this reason, accurate results are far more likely when NARS has been pre-loaded with enough initial data for more complex reasoning to be done initially on. This can be done directly in this project, through a 'loading' feature to load in a series of full statements.

3.2 Speed & Responsiveness

As brought up previously, this project uses localized LLMs through Ollama instead of the use of APIs. While there are many advantages to doing this, it does bring a major limitation of speed. APIs have the full resources of the associated LLM, including large amounts of computing power. By using Ollama models, the response time for the project is limited by the hardware of the user. This makes it noticeably slower, particularly for single users. Researchers or research labs with dedicated computing servers may be less prone to running into this responsiveness limitation.

3.3 Potential Loss of Accuracy

We believe that this project can improve accuracy compared to the use of LLMs alone. This is especially important as the reliance on LLMs in everyday life is something we do not believe is going away, so options to make it more reliable and accurate are key. However, the integration of LLMs into anything, particularly something as fact-based as NARS does bring system accuracy down. This project does likely lose some accuracy in specific circumstances compared to using NARS alone. However, the benefits of being able to use complex natural language, compared to the simple natural language translators that existed within ONA initially, do far outweigh this.

4 Practical Evaluation

This section presents the findings from our empirical evaluation of the NARS-LLM hybrid system across three distinct document types, each of which represents a potential use-case for this system. The experimental design employed a controlled testing methodology to assess the system's performance in extracting, converting, and interpreting factual information. In order to reduce the informational noise from the language models in responses, these documents contain entirely fabricated information.

4.1 Experimental Design

Three document types were selected to evaluate the system's robustness across varying content structures:

1. A conceptual article describing the fictional "Glumboldt Resonance," containing abstract relationships and theoretical concepts.

2. A structured business meeting transcript featuring explicit entity relationships and procedural dialogue

3. A narrative fable presented in two parts, containing concrete entities within a fictional context

Each document was processed one sentence at a time through fact extraction (Minstrel 7B), narsese to english conversion, and interpretation (Deepseek-R1).

4.1.1 Per-Document Performance

Our analysis reveals unexpected performance patterns across document types. The fable achieved the most successful extraction, accurately identifying concrete entities (Edran) and their attributes (horn key, wyrdspire) with coherent output. The Glumboldt Resonance article resulted in moderately successful extraction, though the system resorted to speculative connections when encountering abstract concepts. Surprisingly, the structured business transcript produced the poorest results, with almost entirely garbled output despite its explicit relational structure.

This suggests that there is a weakness in our ability to translate complicated relationships into Narsese, even if we can extract them. The fable contains relatively simple relationships "has", "went to", and "used" are all clearly represented in the text. The Glumboldt Resonance article contains many more nebulous relationships, and the business transcript contains almost none.

4.1.2 Narsese-to-English Conversion Quality

Critical deficiencies were identified in the conversion algorithm's ability to reconstruct coherent English from Narsese representations. Examples from the transcript results include malformed outputs such as "It infrastructure is 's and ovement it is from." Even when Narsese storage maintained semantic validity, the reverse translation frequently produced fragmented or semantically incomplete English statements.

4.1.3 Context Processing and Interpretation

Deepseek-R1 demonstrated varying success when processing unfiltered NARS output. While the Glumboldt Resonance test produced coherent interpretations that aligned with the source material (connecting sound design, harmonic elements, and spatial audio concepts present in the original text), the transcript results suffered from complete breakdown in semantic coherence. This disparity suggests the issue lies not primarily in context overload but in the quality of input received from the conversion layer.

5 Findings

Our empirical evaluation revealed several important insights about the hybrid system's behavior relative to its potential limitations:

5.1 Initial Information Requirements

Contrary to our expectations, the reliance on initial information (Section 3.1) proved to be less of an obstacle than anticipated. When the system successfully extracted relationships with good qualityas seen with the fable's simple entity-attribute pairsit was able to derive relevant information even from a minimal knowledge base. However, as relationship complexity increased, the need for extensive initial data became more pronounced.

5.2 Performance Bottlenecks

The speed limitations (Section 3.2) extended beyond the anticipated LLM constraints. As NARS's knowledge base grew during testing, we observed significant increases in inference step execution time. This compounded the already limited responsiveness of local LLM models. We found that using a smaller model (Mistral AI) specifically for fact extraction helped partially mitigate LLM-related delays.

5.3 Accuracy Gains and Translation Challenges

The system demonstrated marked accuracy improvements compared to standalone LLM performance. Models alone had no knowledge of our fabricated test subjects, yet the hybrid system extracted coherent information from these concepts. This suggests the approach could generalize to real-world topics, though the Narsese-to-English translation bottleneck severely limits practical application for complex content structures.

5.4 Noted Areas for System Improvement

- *Relationship Extraction Deficiency*: The system struggles more with complex relational structures than with simpler entity-attribute relationships.
- *Translation Fidelity*: The Narsese-to-English conversion layer represents a critical failure point, completely breaking down for complex documents like the transcript.
- *Context Management*: The current implementation lacks filtering mechanisms, but this limitation is overshadowed by the translation failures that prevent coherent output.

These findings show that while the NARS-LLM hybrid improves accuracy, the conversion layer between Narsese and English creates a major bottleneck.

Simple relationships like "has" and "went to" work well, but complex relationships break down completely. This translation limitation prevents the current system from handling real-world data effectively.

6 Conclusions

This project demonstrated both the potential and limitations of combining NARS with LLMs to improve reasoning accuracy. We demonstrated that NARS can provide true relational representations of concepts that the system successfully extracts, as evidenced by its ability to interpret and infer relationships from fabricated information. However, the translation layer between natural language and Narsese represents a critical bottleneck that prevents the system from handling real-world complexity.

Future work should prioritize developing robust Narsese-to-English conversion algorithms. With improved translation mechanics, this approach could meaningfully combine the reasoning capabilities of NARS with the natural language understanding offered by modern LLMs.

7 Relevant Works

- **OpenNARS for Applications (ONA)**. The NARS implementation used in this project [1].
- **NARS-GPT**. Earlier implementation demonstrating feasibility of NARS-LLM integration [2].
- **NARS Python**. Alternative Python implementation of NARS, considered for this project [3].
- **NAR-HYBRID**. All test documents, system outputs, and experimental results available in the project repository and attached zip file [4].

References

[1] OpenNARS for Applications. [Online]. Available: `https://github.com/opennars/OpenNARS-for-Applications`

[2] NarsGPT. [Online]. Available: `https://github.com/patham9/NarsGPT`

[3] NARS-Python. [Online]. Available: `https://github.com/ccrock4t/NARS-Python`

[4] Goldmeer, A. and Conley, N. NAR-HYBRID GitHub Repository. [Online]. Available: `https://github.com/gummyfrog/NAR-HYBRID`