

Proceedings of the Second International Workshop on Next-Generation Language Models for Knowledge Representation and Reasoning (NeLaMKRR 2025)

Held in conjunction with the 22nd International Conference on
Principles of Knowledge Representation and Reasoning (KR 2025)

November 12, 2025

Workshop Organizers

- Ken Satoh, Center for Juris-Informatics, ROIS-DS, Tokyo, Japan
- Ha-Thanh Nguyen, Research and Development Center for Large Language Models, NII, Tokyo, Japan
- Francesca Toni, Imperial College London, United Kingdom
- Randy Goebel, Alberta Machine Intelligence Institute, University of Alberta, Canada
- Kostas Stathis, Royal Holloway University of London, United Kingdom

Program Committee

- Agnieszka Mensfelt, Royal Holloway, University of London, United Kingdom
- Daniel Sonntag, German Research Center for Artificial Intelligence (DFKI), Germany
- Gabriel Freedman, Imperial College London, United Kingdom
- John D. Martin, Openmind Research Institute, Edmonton, Alberta, Canada
- Lihu Chen, Imperial College London, United Kingdom
- María Navas-Loro, Universidad Politécnica de Madrid, Spain
- May Myo Zin, Center for Juris-Informatics, ROIS-DS, Tokyo, Japan
- Minh-Phuong Nguyen, Japan Advanced Institute of Science and Technology, Japan
- Sabine Wehnert, Otto-von-Guericke University Magdeburg, Germany
- Thi-Hai-Yen Vuong, VNU University of Engineering and Technology, Vietnam
- Vince Trencsenyi, Royal Holloway University of London, United Kingdom
- Vu Tran, Japan Advanced Institute of Science and Technology, Japan
- Wachara Fungwacharakorn, Center for Juris-Informatics, ROIS-DS, Tokyo, Japan

Introduction

Reasoning is an essential component of human intelligence as it plays a fundamental role in our ability to think critically, support responsible decisions, and solve challenging problems. Traditionally, AI has addressed reasoning in the context of logic-based representations of knowledge. However, the recent leap forward in natural language processing, with the emergence of language models based on transformers, is hinting at the possibility that these models exhibit reasoning abilities, particularly as they grow in size and are trained on more data. Despite ongoing discussions about what reasoning is in language models, it is still not easy to pin down to what extent these models are actually capable of reasoning.

The goal of this workshop is to create a platform for researchers from different disciplines and/or AI perspectives, to explore approaches and techniques with the aim to reconcile reasoning between language models using transformers and using logic-based representations. The specific objectives include analyzing the reasoning abilities of language models measured alongside KR methods, injecting KR-style reasoning abilities into language models (including by neuro-symbolic means), and formalizing the kind of reasoning language models carry out. This exploration aims to uncover how language models can effectively integrate and leverage knowledge and reasoning with it, thus improving their application and utility in areas where precision and reliability are a key requirement.

Workshop Overview

The Second International Workshop on Next-Generation Language Models for Knowledge Representation and Reasoning (NeLaMKRR 2025) brought together researchers from diverse disciplines to explore the intersection of large language models and symbolic reasoning systems. The workshop featured **six long papers** and **four short papers**, covering topics ranging from argumentation and case-based reasoning to ontology construction, autoformalization, legal compliance verification, and moral judgment steering in language models.

The workshop program included presentations of accepted papers, an invited talk by Prof. Hisashi Hayashi on the evolution from classical agents to LLM agents, and interactive discussion sessions. All papers underwent rigorous peer review by at least two program committee members, ensuring high-quality contributions to the field.

Table of Contents

Long Papers

1. **Argumentative Reasoning with Language Models on Non-factorized Case Bases** (Page 4)
W. FunGWacharakorn, M. M. Zin, H.-T. Nguyen, Y. Kong, K. Satoh
2. **Specific Domain Ontology Construction Using Large Language Models** (Page 13)
Vivian Magri Alcadi Soares, Renata Wassermann
3. **Towards a Common Framework for Autoformalization** (Page 23)
A. Mensfelt, D. T. Cucala, S. Franco, A. Koutsoukou-Argraki, V. Trencsenyi, K. Stathis
4. **Multi-Agent Legal Verifier Systems for Data Transfer Planning** (Page 35)
H.-T. Nguyen, W. FunGWacharakorn, K. Satoh
5. **HA-GNN: Learning Acyclic Hierarchies in Scientific Text with Hierarchy-Aware Graph Neural Networks** (Page 43)
D. Joshi, I. Rekik
6. **Which Neurons Nudge Moral Judgments? Neuron-Level Last-Token Steering in Large Language Models** (Page 50)
D. Liga, L. Yu, R. Markovich

Short Papers

7. **Applying Relation Extraction and Graph Matching to Answering Multiple Choice Questions** (Page 57)
N. Shimoda, A. Yamamoto
8. **FC-CONAN: An Exhaustively Paired Dataset for Robust Evaluation of Retrieval Systems** (Page 63)
J. Junqueras, F. Boudin, M. M. Zin, H.-T. Nguyen, W. FunGWacharakorn, D. A. Furman, A. Aizawa, K. Satoh
9. **On the Role of Domain Experts in LLM-based Knowledge Formalization** (Page 70)
S. Vandeveld
10. **From Data Logs to Narrative Intelligence: Integrating ASP Logic and LLMs for Multi-robot Mission** (Page 76)
S. Madasamy, K. Sammut, R. Leibbrandt, P. Santos, C. Buche, V. Martin, A. G. Bossier

Argumentative Reasoning with Language Models on Non-factorized Case Bases

Wachara Fungwacharakorn¹, May Myo Zin¹, Ha Thanh Nguyen^{1,2}, Yuntao Kong¹, Ken Satoh¹

¹ Center of Juris-Informatics, Joint Support-Center for Data Science Research, ROIS, Tokyo, Japan

² Research and Development Center for Large Language Models, NII, ROIS, Tokyo, Japan

{wacharaf, maymyozin, nguyenhathanh, kongyt, ksatoth}@nii.ac.jp

Abstract

In this paper, we investigate how language models can perform case-based reasoning (CBR) on non-factorized case bases. We introduce a novel framework, argumentative agentic models for case-based reasoning (AAM-CBR), which extends abstract argumentation for case-based reasoning (AA-CBR). Unlike traditional approaches that require factorization of previous cases, AAM-CBR leverages language models to determine case coverage and extract factors based on new cases. This enables factor-based reasoning without exposing or preprocessing previous cases, thus improving both flexibility and privacy. We also present initial experiments to assess AAM-CBR performance by comparing the proposed framework with a baseline that uses a single-prompt approach to incorporate both new and previous cases. The experiments are conducted based on a synthetic credit card application dataset. The result shows that AAM-CBR surpasses the baseline only when the new case contains a richer set of factors. The finding indicates that language models can handle case-based reasoning with a limited number of factors, but face challenges as the number of factors increase. Consequently, integrating symbolic reasoning with language models, as implemented in AAM-CBR, is crucial for effectively handling cases involving many factors.

1 Introduction

Case-based reasoning (CBR) is a classic reasoning task in artificial intelligence (AI), widely applied in domains such as law, finance, and healthcare. In CBR, decisions for new cases are derived by drawing analogies to previous cases with known outcomes. One major representation of cases in CBR is a set of *factors* (Aleven and Ashley 1995). Each factor captures a significant dimension that influences the outcome. These factors are annotated and serve as abstracted features that allow analogical comparisons between cases. Reasoning proceeds by identifying similarities and differences in the factor sets of the current and previous cases, and applying heuristics or argumentation approaches to reach the outcome for a new case.

Although factor-based representations support structured and interpretable reasoning, several limitations remain. One limitation is the impossibility of qualifying all factors in the first place, as new cases can lead to new factors. Another limitation is the significant effort to identify factors in previous cases, as the reasoning usually considers a large number

of previous cases, and each previous case is described in a very long text. In addition to that, sometimes it is necessary to identify the magnitudes or dimensions of factors, as they can affect the reasoning.

This paper raises an alternative question: can we use language models for case-based reasoning without initially factorizing previous cases, but the reasons can still be interpretable as factor-based reasoning. To reduce the scope of the question, this paper focuses particularly on **abstract argumentation for case-based reasoning** (AA-CBR) (Çyras, Satoh, and Toni 2016). Originally, AA-CBR requires factorizing previous cases to determine relevance, especially against new cases. However, in this paper, we extend AA-CBR into a novel framework, **argumentative agentic models for case-based reasoning** (AAM-CBR), which instead takes advantage of language models to determine relevance. Therefore, the proposed framework does not require factorizing previous cases. Only new cases are factorized and used to determine relevance by language model agents attached to previous cases. This benefits from propagating new factors from new cases, as well as keeping the information in the previous case private. In addition, this paper presents initial experiments on whether AAM-CBR, which is partially black-boxed in case coverage determination and case factor extraction, can perform better than putting previous cases and new cases in a single prompt, which is solely black-boxed, in predicting the AA-CBR outcome. The experiments are grounded in the domain of credit card application decisions. The result shows that AAM-CBR performs better than the single-prompt approach only when new cases cover more factors and struggles when new cases cover fewer factors. This highlights the importance of integrating symbolic reasoning with language models, as implemented in AAM-CBR, especially when the number of factors increases.

The paper is structured as follows. Section 2 provides related work on the background of case-based reasoning, abstract argumentation, and largelanguage models. Section 3 provides the background of abstract argumentation for case-based reasoning (AA-CBR). Section 4 presents the proposed framework, argumentative agentic models for case-based reasoning (AAM-CBR). Section 5 describes the experiments conducted for this paper. Section 6 presents the results of the experiments. Section 7 discusses the results and suggests future work. Finally, Section 8 concludes this paper.

2 Related Work

This section provides the backgrounds of argumentative reasoning, case-based reasoning, and large language models, positioning our AAM-CBR framework within those backgrounds.

2.1 Argumentation and Case-based Reasoning

Since the foundation of abstract argumentation framework (Dung 1995), there have been numerous computational argumentation extensions that address practical reasoning scenarios. Dung, Mancarella, and Toni (2002) provided proof procedures for credulous and sceptical nonmonotonic reasoning, establishing computational foundations that enable practical reasoning implementations. Bench-Capon et al. (2003) explored the integration of computational argumentation with legal practice, providing insights into how formal argumentation can support real-world legal reasoning.

In particular, several researchers have focused on integrating argumentative reasoning with case-based reasoning (CBR). Prakken et al. (2015) formalized argumentation schemes for legal case-based reasoning within the ASPIC+ framework, demonstrating how structured legal reasoning can be systematically captured through argument-based models. Ćyras et al. (2016) introduced a formal framework bridging these two paradigms, allowing CBR to leverage the inferential structure and evaluation semantics of abstract argumentation. In addition to this, Al Abdulkarim (2017) proposed methods to represent case law in a form suitable for argumentative reasoning, with a focus on legal applications. These contributions collectively underscore the potential of combining computational argumentation and CBR to enhance the explainability, structure, and normative grounding of AI legal reasoning systems (Atkinson, Bench-Capon, and Bollegala 2020).

The relevance of computational argumentation has been further explored in the context of AI systems that require explainability and contestability. Rotolo and Sartor (2023) investigated computational argumentation and explanation in law, connecting between formal justification and explanation. Leofante et al. (2024) proposed the necessity of computational argumentation for building contestable AI systems, advocating for structured reasoning mechanisms that can justify and challenge AI decisions in complex environments.

2.2 Neural and Hybrid Case-Based Reasoning

The integration of neural networks with traditional CBR has been long investigated. Early work by Ardit and Tokdemir (1999) conducted empirical comparisons in engineering domains, demonstrating that neural networks excel at pattern recognition while CBR provides better interpretability. Chen and Burrell (2001) provided a systematic comparison between case-based reasoning systems and artificial neural networks, identifying complementary strengths that could be leveraged through hybrid approaches.

Recent advances have focused on deep learning integration. Li et al. (2018) introduced prototype-based neural networks that perform case-based reasoning through learned

prototypes, achieving both accuracy and interpretability. The work demonstrates how neural networks can learn meaningful case representations while maintaining the explanatory power of traditional CBR. Amin et al. (2018) proposed a hybrid system combining deep neural networks with CBR for complex reasoning tasks, showing how deep learning can enhance case retrieval and adaptation processes. More recently, Gould and Toni (2025) introduced Gradual Abstract Argumentation for Case-Based Reasoning (Gradual AA-CBR), a neuro-symbolic model utilizing case-based reasoning as an end-to-end method, further advancing the goal of interpretable and structured neural case-based reasoning.

Meanwhile, recent research has explored the potential of large language models (LLMs) in addressing limitations of factor-based reasoning in CBR. Gray et al. (2023) demonstrated the use of LLMs for automatic identification of relevant factors from textual case descriptions. Subsequent work (Gray et al. 2024) investigated how LLMs can be employed to discover novel factors from case corpora, expanding the representational capacity of traditional approaches, and further research by the same group focused on estimating the magnitudes of such factors using LLMs (Gray et al. 2025). These efforts highlight the growing role of LLMs in augmenting core CBR processes such as case representation and factor extraction, pushing the boundaries of hybrid reasoning systems.

2.3 Large Language Models for Reasoning

The application of large language models to reasoning tasks has emerged as a major research direction. Huang and Chang (2022) provided an early survey of reasoning capabilities in LLMs, identifying key challenges and opportunities. Plaat et al. (2024) offered a more recent perspective, examining how reasoning abilities have evolved with larger and more sophisticated models.

In domain-specific applications, particularly in law, logical reasoning reliability and interpretability are critical, and LLMs still struggle with such reasoning capabilities. Nguyen et al. (2023a) evaluated how well state-of-the-art legal reasoning models support abductive reasoning, a key capability for forming explanatory hypotheses from observations. Despite strong performance in certain legal tasks, current models still struggle with abductive inference. Similarly, Nguyen et al. (2023b) assessed negation detection in GPT models and found that even the strongest models, like GPT-4 at that time, face notable challenges. These results underscore that key aspects of logical reasoning, such as negation handling and hypothesis generation, remain unresolved.

To enhance LLM reasoning capabilities, multi-step reasoning has received particular attention. Aksitov et al. (2023) developed self-improvement methods for multi-step reasoning in LLM agents, combining reinforcement learning with trajectory optimization. Wang et al. (2024) introduced the Q* framework for improving multi-step reasoning through deliberative planning, addressing pathologies in sequential reasoning processes.

Argumentative reasoning with LLMs has also attracted

growing attention. Castagna, Sassoon, and Parsons (2024) proposed enhancing LLM reasoning by incorporating critical questions inspired by Toulmin’s model of argumentation (Toulmin 2003), aiming to improve the coherence and structure of model-generated arguments. Sukpanichnant, Rapberger, and Toni (2024) presented *PeerArg*, an argumentative reasoning framework using LLMs to support scientific peer review. Liga, Markovich, and Yu (2025) introduced a hybrid approach that combines abstract argumentation with LLMs through prompt engineering, specifically designed to address legal requirements such as the right to explanation and the right to challenge. Freedman et al. (2025) introduced argumentative LLMs (*ArgLLMs*), a method to augment LLMs with argumentative reasoning to enhance the explainability of LLMs and allow users to contest LLMs to correct mistakes. Collectively, these approaches illustrate the growing potential of combining LLMs with computational argumentation to enhance explainability and contestability in generated reasoning.

3 AA-CBR

This section provides the background on **abstract argumentation for case-based reasoning** (AA-CBR) (Čyras, Satoh, and Toni 2016). In AA-CBR, a previous case is considered as a pair of a finite set of factors and their decided outcome. Let \mathbb{F} be a set of all possible factors. Each subset of \mathbb{F} is called a *situation*. AA-CBR assumes a binary distribution of outcomes, which we assume as a set $\mathbb{O} = \{0, 1\}$. A *previous case* is now a pair of $(X, o) \in 2^{\mathbb{F}} \times \mathbb{O}$. A *case base* Γ is a finite set of outcome-consistent cases (i.e., for $(X, o_x), (Y, o_y) \in \Gamma$, if $X = Y$, then $o_x = o_y$).

AA-CBR uses an abstract argumentation framework (AA framework) (Dung 1995), which we recap as follows. AA framework is a pair $(\mathcal{A}, \rightsquigarrow)$. Each element of \mathcal{A} represents an *argument* and \rightsquigarrow is a binary relation over \mathcal{A} representing *attacks* between arguments. For $x, y \in \mathcal{A}$, if $x \rightsquigarrow y$ then we say x attacks y . For a set of arguments $E \subseteq \mathcal{A}$ and an argument $x \in \mathcal{A}$, E defends x if, for every $y \in \mathcal{A}$ that attacks x , there is an argument $z \in E$ that attacks y . Then, the *grounded extension* of $(\mathcal{A}, \rightsquigarrow)$ can be constructed inductively as $G = \bigcup_{i \geq 0} G_i$, where G_0 is the set of unattacked arguments, and for $i \geq 0$, G_{i+1} is the set of arguments that G_i defends.

AA-CBR needs to assume a *default* outcome $o_d \in \mathbb{O}$, which is inferred as an outcome for the empty case. \bar{o}_d represents the contrary one (i.e., $\bar{o}_d \in \mathbb{O} \setminus \{o_d\}$). Given a case base Γ , an AA framework corresponding to Γ , a default outcome $o_d \in \mathbb{O}$, and a new case $N \subseteq \mathcal{F}$ is $(\mathcal{A}, \rightsquigarrow)$ satisfying the following conditions (Čyras, Satoh, and Toni 2016):

1. (*arguments*) $\mathcal{A} = \Gamma \cup \{(N, ?)\} \cup \{(\emptyset, o_d)\}$;
2. (*case attacks*) for $(X, o_x), (Y, o_y) \in \Gamma \cup \{(\emptyset, o_d)\}$, it holds that $(X, o_x) \rightsquigarrow (Y, o_y)$ iff
 - (different outcomes) $o_x \neq o_y$, and
 - (specificity) $Y \subsetneq X$, and
 - (concision) $\nexists (Z, o_z) \in \Gamma$ with $Y \subsetneq Z \subsetneq X$;
3. (*irrelevant attacks*) for $(Y, o_y) \in \Gamma$, $(N, ?) \rightsquigarrow (Y, o_y)$ holds iff $Y \not\subseteq N$.

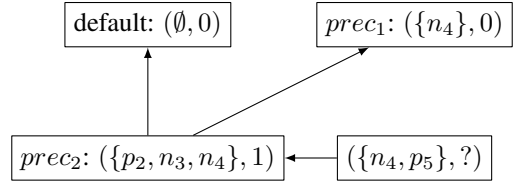


Figure 1: Corresponding AA framework with default outcome ‘0’

The AA-CBR outcome of the new case N is

- the default outcome o_d if (\emptyset, o_d) is in the grounded extension of the corresponding AA framework;
- the contrary \bar{o}_d , otherwise.

Throughout this paper, we consider the domain of credit card application decisions. The outcome ‘0’ represents the rejection of the credit card application and ‘1’ represents the approval of the credit card application. We consider the following set \mathbb{F}_{credit} of these factors:

1. p_1 : low debt-to-income ratio
2. p_2 : long and stable employment history
3. p_3 : consistent payment history on existing loans
4. p_4 : significant assets declared
5. p_5 : positive relationship with the bank
6. n_1 : high number of recent credit inquiries.
7. n_2 : missed or late payments history
8. n_3 : insufficient income
9. n_4 : limited credit history
10. n_5 : young age

Factor 1-5 (represented as p_1 - p_5) are *positive* factors as they favour the approval of the application. Meanwhile, factor 6-10 (represented as n_1 - n_5) are *negative* factors as they favour toward the rejection of the application. Please note that some CBR models, such as precedential constraint (Horty 2004), consider these favours in their reasoning approach, but not AA-CBR. We just distinguish them here to make cases realistic (for detailed comparison between AA-CBR and precedential constraint, see Paulino-Passos and Toni (2021) and Fungwacharakorn et al. (2025)).

Suppose we have the following previous cases:

1. Case 1: the applicant had a limited credit history (n_4). This credit card application was rejected.
2. Case 2: The applicant had an insufficient income (n_3) and a limited credit history (n_4), but had a long and stable employment history (p_2). This credit card application was approved.

We can then represent these previous cases as the following case base:

$$\Gamma_1 = \{(\{n_4\}, 0), (\{n_3, n_4, p_2\}, 1)\}.$$

Suppose that we have a new application where the applicant has a limited credit history (n_4) but has a positive

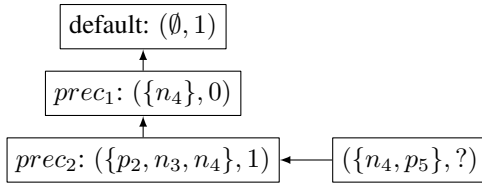


Figure 2: Corresponding AA framework with default outcome ‘1’

relationship with the bank (p_5). This application can be represented as $N_1 = \{n_4, p_5\}$. If we consider ‘0’ as a default outcome, then the AA framework corresponding to $\Gamma_1, '0'$, and N_1 can be depicted in Figure 1, where $(\emptyset, 0)$ is in the grounded extension of the corresponding framework (the grounded extension is $\{(\emptyset, 0), (\{n_4\}, 0), (\{n_4, p_5\}, ?)\}$). Thus, the AA-CBR outcome of N_1 is ‘0’. Meanwhile, if we consider ‘1’ as a default outcome, then the AA framework corresponding to $\Gamma_1, '1'$, and N_1 can be depicted in Figure 2, where $(\emptyset, 1)$ is not in the grounded extension of the corresponding framework (the grounded extension is $\{(\{n_4\}, 0), (\{n_4, p_5\}, ?)\}$). Thus, the AA-CBR outcome of N_1 is also ‘0’, meaning that the new application should be rejected.

4 Proposed Framework

This section introduces **argumentative agentic models for case-based reasoning** (AAM-CBR), which enables factor-based reasoning on non-factorized case bases. AAM-CBR is designed to operate within the abstract argumentation semantics of AA-CBR while minimizing the need for preprocessing of previous cases. To achieve this, the framework orchestrates language model agents, each of which is assigned to one previous case. Figure 3 provides an overview of the AAM-CBR architecture. The reasoning pipeline consists of three primary components: (1) case coverage determination, (2) case factor extraction, and (3) AA-CBR based outcome prediction.

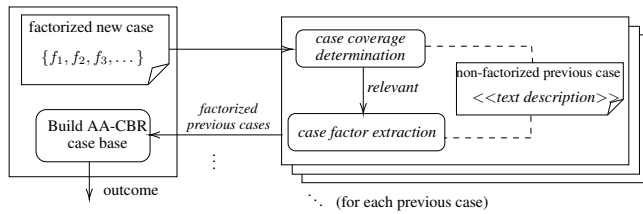


Figure 3: Overview of AAM-CBR

Given a new case represented as a structured set of factors, the AAM-CBR framework determines the appropriate outcome by using a collection of previous cases provided in unstructured natural language form without prior factorization.

For each agent attached to a previous case, the process begins with the *case coverage determination* module, where a language model is used to determine whether the factors of the new case cover the situation described in the textual

description. If the factors cover the situation, then the previous case is determined as *relevant* according to AA-CBR. This is similar to *case coverage* in factor-based reasoning (Aleven and Ashley 1995; Sartor 2002). For the previous case that is deemed “*relevant*”, the agent will proceed to the *case factor extraction* module. In this module, a language model is used to determine which factors in the new case are implied in the situation of the previous case. The output of the second module is a factorized previous case, composed of factors from the new case implied by the situation in the previous case. For the previous case that is not deemed relevant, however, the agent will refuse the response from the new case agent in order to maintain privacy. This is performed in parallel for every previous case in the case base. The factorized previous cases are then used to build an AA-CBR case base to predict an outcome. Therefore, if the case coverage determination and the case factor extraction work perfectly, the framework will predict the same outcome as AA-CBR.

5 Experimental Setup

This section describes the experiments used in this paper. The experiments are grounded in the domain of credit card application decisions, based on the same set \mathbb{F}_{credit} of factors shown in Section 3. The experiments include the steps of generating scenarios, generating test sets, determining case coverage, extracting case factors, and predicting the outcome. The details of each step are as follows.

5.1 Generating Scenarios

The first step is to enumerate all subsets of \mathbb{F}_{credit} and generate scenarios. A scenario is defined as an example textual description of situation that can be represented by the factors in the subset under consideration and not those outside the subset. To generate a scenario, we use the following prompt template:

Prompt 1: Generating scenarios

TASK:

Your task is to generate an example of credit card application scenario that covers a specified set of factors and excludes another specified set of factors.

INSTRUCTIONS:

You will be provided with a specified set of factors that should be covered in the generated scenario and another specified set of factors that should NOT be covered in the generated scenario.

The set of factors that should be covered in the generated scenario:

{included_factor_list}

The set of factors that should NOT be covered in the generated scenario:

{excluded_factor_list}

OUTPUT FORMATTING: Generate the scenario in one concise description. Do NOT explicitly use the same words as those in factors. Do NOT include an outcome

whether the credit card is accepted or rejected.

To check whether the scenario considers only the factors in the subset, a language model is then used to extract the factors back from the scenario description, with the following prompt template:

Prompt 2: Extracting factors

TASK:
Your task is to extract factors from a description of a credit card application scenario.
INSTRUCTIONS:
You will be provided with a description of a credit card application scenario and a list of all possible factor sentences.
Description: {description}
All possible factors: {all_factor_sentences}
Identify and return ONLY the factor sentences from the provided list that are explicitly present or clearly implied in the description.
OUTPUT FORMATTING: a JSON array of the extracted factor sentences. If no factors are found, return [].

If the set of extracted factors is the same as the original subset, then we keep that scenario. Otherwise, we tried to generate a new scenario corresponding to that subset upto 10 times. If it still fails after 10 times, then we just skip that subset. Since \mathbb{F}_{credit} has 10 factors, $2^{10} - 1 = 1023$ scenarios can be generated at most from the non-empty subsets of \mathbb{F}_{credit} . However, since some subsets are skipped, the number of actual scenarios is slightly lower than 1023. Here is one example of generated scenario, for a subset $\{p_1, n_2\}$:

Sarah, a middle-aged individual, applies for a new credit card. Her monthly expenses are manageable compared to her earnings, resulting in a favorable debt-to-earnings ratio. However, her record includes a few instances of overdue bills from years past. While not excessively burdensome, these past indiscretions are recorded on her credit report. She is applying to a new financial institution.

5.2 Generating Test Sets

Next, we generate test sets based on the generated scenario. Each test set contains 10 assumed previous cases and 5 new cases. Each previous case is randomly selected from the scenarios and assigned either an outcome '0' (the credit card application was rejected) or '1' (the credit card application was approved), with the following constraints:

1. If the selected scenario is generated from the subset with merely negative factors (resp. positive factors), then the outcome must be '0' (resp. '1').
2. To maintain the outcome consistency, if the scenario has already been selected as a previous case in the same test set, then it is assigned with the same outcome.

Meanwhile, a new case is merely a subset of \mathbb{F}_{credit} with no outcome assigned. Each test set contains 5 new cases,

each having 6, 7, 8, 9 and 10 factors (the 10-factor new case is then unique and identical to \mathbb{F}_{credit}).

5.3 Determining Case Coverage

Then, we conducted an experiment on the first module of AAM-CBR: case coverage determination. The experiment considers each pair of previous and new cases (hence, $10 \times 5 = 50$ pairs in each test set). Each pair is considered whether or not the factors in the new case cover the situation in previous case, which is counted as *relevant* in AA-CBR. Unlike the original AA-CBR, we instruct the LLM to determine case coverage by the case description from the previous case and the factors of the new case instead, with the following prompt template:

Prompt 3: Determining case coverage

TASK:
Your task is to determine whether the factor list covers the case.
INSTRUCTIONS:
You will be provided with a factor list and a case description.
Here is the factor list:
{factor_list}
Here is the case description:
{case_description}
Answer this question: does the factor list cover the case?
OUTPUT FORMATTING: 'YES' or 'NO'

5.4 Extracting Case Factors

After that, we conducted an experiment on the second module of AAM-CBR: case factor extraction. The experiment considers each pair of previous and new cases that have been determined as *relevant* by the first module. In this experiment, we reuse Prompt 2 to extract factors. Hence, the experiment becomes a stability evaluation of the case factor extraction prompt. The difference is that the prompt in the scenario generation is fed by the list of all factors in \mathbb{F}_{credit} while the prompt in this experiment is fed by the factors in the new case (which should return similar results if the previous case is actually relevant to the new case i.e., the factors in the new case cover the previous case).

5.5 Predicting Outcomes

After that, we conducted the main experiment, that is, to evaluate the performance of AAM-CBR in predicting the AA-CBR outcome. Since AA-CBR needs to assume a default outcome, we divide the gold standard for the prediction into two modes, one for the default outcome '0' and another for the default outcome '1'. The prediction considers the set of all previous cases in each test set as a case base and considers an individual new case in each test set. We use a baseline method, called a *SinglePrompt* prediction, which involves feeding all previous cases and new cases into a single prompt to predict an outcome. A *SinglePrompt* prediction is divided into two options according to its use of argumentative reasoning structure. The first option, referred to as *non-instructed*, does not provide any argumentative reasoning structure, as the following prompt template:

Prompt 4: Predicting outcome (non-instructed)

TASK:

You are an expert Case-Based Reasoning (CBR) system. Your task is to predict the outcome for a new case based on given previous cases.

INSTRUCTIONS:

You will be provided with previous cases, a new case, and a default outcome.

Here are the previous cases you will be working with.

{previous_case_list}

And this is the new case to analyze:

{new_case_list}

The default outcome is '{default_outcome}'

Based on the previous cases provided, what is the most likely outcome for this new case?

The second option, referred to as *instructed*, provides the argumentative approach based on the dispute trees in AA-CBR. It extends the previous prompt template by replacing the last paragraph in the template with the following instructions:

Prompt 5: Predicting outcome (instructed-revision)

Please do NOT consider the positiveness and negativeness of factors.

You will use a dialectical process between a proponent and an opponent.

Your decision-making process should follow these steps:

1. The proponent starts by asserting a default claim with empty factors and the default outcome '{default_outcome}'
2. The opponent can challenge the proponent's claim if they can identify a previous case that meets the following criteria:
 - with all factors covered by the new case
 - with the outcome '{opponent_outcome}'
 - If the challenging previous case's factors are NOT covered by the proponent's claimed case, then the opponent CANNOT challenge with this previous case (this is a STRICT condition).
3. The proponent can defend against the opponent's rebuttal if they can identify a previous case that meets the following criteria:
 - with all factors covered by the new case
 - with the outcome '{default_outcome}'
 - If the defending previous case's factors are NOT covered by the opponent's claimed case, then the proponent CANNOT defend with this previous case (this is a STRICT condition).
4. After considering all possible argumentative paths:
 - If the proponent cannot uphold their initial claim through this process (meaning there is no winning path for them), then the predicted outcome for

the new case will be '{opponent_outcome}'

- Otherwise, the predicted outcome for the new case will be '{default_outcome}'

According to the steps provided, what is the predicted outcome for the new case?

We found that the prediction works better if we allow the language model to respond with an explanation (see Section 7). Therefore, we introduce two-step prompting by letting a language model respond with an explanation first, then extracting the prediction from the first response using the following prompt template:

Prompt 6: Concluding predicted outcome

TASK: Your task is to conclude the predicted outcome from the response.

INSTRUCTIONS: Here is the response:

{first_response}

What is the predicted outcome from this response (answer 'mixed' if the predicted outcome cannot be concluded)?

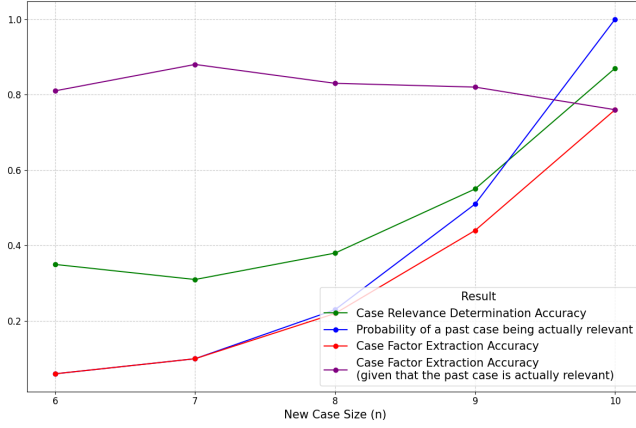
OUTPUT FORMATTING: '{outcome0}' or '{outcome1}' or 'mixed'.

For AAM-CBR, we use the results from the second module: case factor extraction, to build an AA-CBR case base to predict the AA-CBR outcome. Therefore, we do not need to instruct a language model additionally.

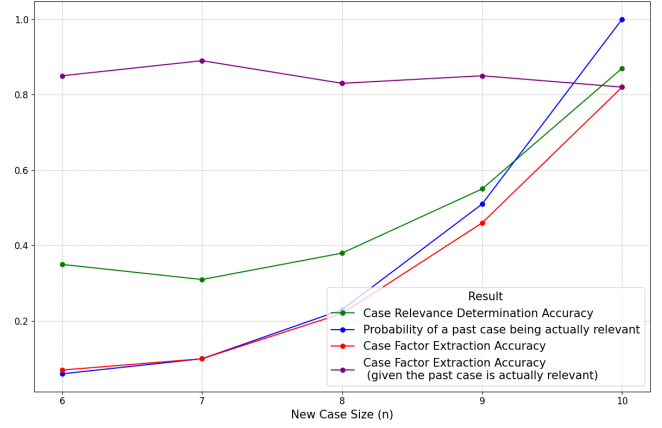
6 Experimental Result

This section presents the results from the initial experiments to evaluate the AAM-CBR framework based on a synthetic dataset that simulates credit card application decisions. The experiments focus on three core components: case coverage determination, case factor extraction, and outcome prediction. To facilitate comparative evaluation, we assess the performance of AAM-CBR against two single-prompt baseline models, which vary in their use of argumentative reasoning structure. Using 50 synthetic test sets, evaluations were conducted on each test set, which contains 10 previous cases and 5 new cases of increasing factor richness ($n = 6$ to $n = 10$). We used both GEMINI-2.0-FLASH-LITE and GPT-4o as the underlying language models across all tasks, applying identical inputs and prompts to ensure a fair comparison.

Our first experiment focuses on case relevance determination and factor extraction, varying the number of factors (n) in each new case from 6 to 10. As shown in Figure 4, we report four metrics: (1) *Case Relevance Determination Accuracy*, (2) *Probability that a Retrieved Case is Actually Relevant*, (3) *Case Factor Extraction Accuracy*, and (4) *Factor Extraction Accuracy Given Relevance*. Both models show steady improvements as n increases, with the most significant gains between $n = 9$ and $n = 10$. While performance trends are closely aligned, GPT-4o consistently achieves slightly higher accuracy in factor extraction, particularly when conditioned on relevance. These results suggest a modest but consistent advantage for GPT-4o in retrieval and structured information extraction tasks.



(a) GEMINI-2.0-FLASH-LITE



(b) GPT-4o

Figure 4: Comparative performance on case relevance determination and case factor extraction tasks

We then evaluated the final predicted outcomes for each new case using AAM-CBR and two single-prompt baselines. The baselines differ in whether they were explicitly guided to use argumentative reasoning:

- SinglePrompt-NotInstructed: No guidance on argumentative structure was provided.
- SinglePrompt-Instructed: Includes explicit prompting for dispute-tree style argumentation.
- AAM-CBR: Uses structured, dynamically factorized previous cases based on AA-CBR reasoning.

Tables 1 and 2 show the outcome prediction accuracy for GEMINI-2.0-FLASH-LITE and GPT-4o, respectively. Each table shows prediction accuracy split by the new case size (n) and the default outcome label ('0' or '1').

In both models, AAM-CBR underperforms the single-prompt baselines at smaller new cases. For example, in Table 1, SinglePrompt-NotInstructed achieves an accuracy of 0.76 for outcome '0' at $n = 6$, while AAM-CBR achieves only 0.28. However, AAM-CBR's performance improves significantly as the number of factors increases. At $n = 10$, AAM-CBR achieves 1.00 for outcome '0' and 0.96 for outcome '1', outperforming both single-prompt baselines by a wide margin.

A similar pattern is observed in Table 2 for GPT-4o. While the single-prompt baselines initially perform better at $n = 6$, AAM-CBR surpasses them from $n = 8$ onward. By $n = 10$, AAM-CBR achieves 0.98 for outcome '0' and 0.96 for outcome '1', whereas the best baseline reaches only 0.80 for outcome '1'.

These results indicate that structured retrieval and reasoning, as implemented in AAM-CBR, become increasingly important as new cases contain more factors. In contrast, single-prompt methods struggle to maintain performance as the number of factors increases.

7 Discussion

This section discusses several findings of this paper. The first finding is that symbolic reasoning gains importance as

the number of factors increases. This aligns with other hybrid case-based reasoning research that demonstrates the necessity of a symbolic framework to improve interpretability, especially when reasoning involves complex interactions among a large number of cases. The second finding is that, although AAM-CBR identifies factors only through new cases, increasing the number of known factors still leads to significant improvements in accuracy. This reflects the capability of LLMs to discover new factors, as the problem of case coverage determination can be reduced to the problem of discovering a new factor (i.e., if we cannot discover a new factor, then the given factors already cover the given case). This finding points to a promising direction for future work: developing methods to incrementally learn new factors through a series of cases.

On the other hand, the experiments also reveal that, without cooperating with symbolic frameworks, large language models still struggle with case-based reasoning. When conducting the experiment, the baseline *SinglePrompt* occasionally predicts an outcome using alternative approaches, such as counting positive and negative factors, even with the *instructed* option, where we explicitly instruct LLMs not to consider the positive and negative nature of factors. Furthermore, the experiments also reveal that the predictions without explicit reasoning are less accurate compared to those with explicit reasoning. The finding is consistent with previous studies on multi-step reasoning and leads us to use two-step prompting to increase the accuracy of the prediction and the stability of the language model response. Specifically, the experiment on the *case coverage determination* module indicates a bias toward deeming previous cases relevant to new cases, with precision decreasing exponentially as the number of factors in the new case decreases. The decrement follows from the probability that a previous case is actually relevant, which can be calculated as the probability that a subset of \mathbb{F} is also a subset of a new case $N \subseteq \mathbb{F}$ (size n), which is equal to $2^{(n-|\mathbb{F}|)}$. This probability decreases exponentially. For example, given $|\mathbb{F}| = 10$ as in our setting, $n = 8$ gives the probability $2^{(8-10)} = 25\%$

Table 1: Outcome Prediction Accuracy (GEMINI-2.0-FLASH-LITE)

New case size (n) 'default outcome'	n = 6		n = 7		n = 8		n = 9		n = 10	
	'0'	'1'	'0'	'1'	'0'	'1'	'0'	'1'	'0'	'1'
SinglePrompt-NotInstructed	0.76	0.42	0.62	0.56	0.34	0.66	0.14	0.78	0.00	0.90
SinglePrompt-Instructed	0.56	0.58	0.64	0.68	0.44	0.54	0.40	0.36	0.40	0.48
AAM-CBR	0.28	0.40	0.42	0.40	0.70	0.62	0.88	0.82	1.00	0.96

Table 2: Outcome Prediction Accuracy (GPT-4o)

New case size (n) 'default outcome'	n = 6		n = 7		n = 8		n = 9		n = 10	
	'0'	'1'	'0'	'1'	'0'	'1'	'0'	'1'	'0'	'1'
SinglePrompt-NotInstructed	0.54	0.60	0.56	0.76	0.46	0.62	0.36	0.60	0.18	0.80
SinglePrompt-Instructed	0.64	0.54	0.66	0.64	0.52	0.48	0.22	0.30	0.28	0.44
AAM-CBR	0.24	0.46	0.42	0.48	0.74	0.66	0.84	0.84	0.98	0.96

while $n = 6$ gives the probability $2^{(6-10)} = 6.25\%$. Furthermore, the experiment on the *case extraction* module reveals that large language models still produce unstable responses, resulting in approximately 0.80 accuracy given that the previous case is actually relevant, even when we reuse the same prompt template for both the 'generating scenario' and 'extracting case factors' steps. Thus, it still suggests that prompt engineering is required to reduce biases and ensure stability in both modules to realize the full potential of AAM-CBR.

8 Conclusion

This paper presents argumentative agentic models for case-based reasoning (AAM-CBR), a novel framework that utilizes language models to perform case-based reasoning without requiring factorization of previous cases. AAM-CBR leverages language models to dynamically determine case coverage and extract factors directly from new cases, thereby enhancing both flexibility and privacy by not exposing previous cases. Our experiments, conducted in the domain of credit card application decisions, focused on evaluating the core modules of AAM-CBR: *case coverage determination* and *case factor extraction*, as well as its overall performance in predicting outcomes. The results demonstrated that AAM-CBR significantly outperforms single-prompt baselines when the new case contains a richer set of factors. The findings underscore the importance of integrating symbolic reasoning with large language models, especially when the number of factors increases. The variation in performance due to the richness of factors in new cases highlights two main challenges for future development. The first challenge is to enhance the language model's capability to accurately identify case coverage and extract relevant factors, particularly when new cases are less comprehensive. The second challenge is to improve the overall prediction by discovering and learning factors from a series of cases.

Acknowledgments

This work was supported by the "R&D Hub Aimed at Ensuring Transparency and Reliability of Generative AI Models" project of the Ministry of Education, Culture,

Sports, Science and Technology, the "Strategic Research Projects" grant from ROIS (Research Organization of Information and Systems), and JSPS KAKENHI Grant Numbers, JP22H00543.

References

- Aksitov, R.; Miryoosefi, S.; Li, Z.; Li, D.; Babayan, S.; Kopparapu, K.; Fisher, Z.; Guo, R.; Prakash, S.; Srinivasan, P.; et al. 2023. Rest meets react: Self-improvement for multi-step reasoning LLM agent. *arXiv preprint arXiv:2312.10003*.
- Al Abdulkarim, L. M. 2017. *Representation of case law for argumentative reasoning*. Ph.D. Dissertation, The University of Liverpool (United Kingdom).
- Aleven, V., and Ashley, K. D. 1995. Doing things with factors. In *Proceedings of the 5th international conference on artificial intelligence and law*, 31–41.
- Amin, K.; Kapetanakis, S.; Althoff, K.-D.; Dengel, A.; and Petridis, M. 2018. Answering with cases: a cbr approach to deep learning. In *International Conference on Case-Based Reasoning*, 15–27. Springer.
- Arditi, D., and Tokdemir, O. B. 1999. Comparison of case-based reasoning and artificial neural networks. *Journal of computing in civil engineering* 13(3):162–169.
- Atkinson, K.; Bench-Capon, T.; and Bollegala, D. 2020. Explanation in ai and law: Past, present and future. *Artificial Intelligence* 289:103387.
- Bench-Capon, T.; Freeman, J. B.; Hohmann, H.; and Prakken, H. 2003. Computational models, argumentation theories and legal practice. In *Argumentation machines: New frontiers in argument and computation*. Springer. 85–120.
- Castagna, F.; Sassoon, I.; and Parsons, S. 2024. Critical-questions-of-thought: Steering LLM reasoning with argumentative querying.
- Chen, D., and Burrell, P. 2001. Case-based reasoning system and artificial neural networks: A review. *Neural Computing & Applications* 10(3):264–276.
- Dung, P. M.; Mancarella, P.; and Toni, F. 2002. Argumentation-based proof procedures for credulous and

- sceptical non-monotonic reasoning. In *Computational Logic: Logic Programming and Beyond: Essays in Honour of Robert A. Kowalski Part II*. Springer. 289–310.
- Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial intelligence* 77(2):321–357.
- Freedman, G.; Dejl, A.; Gorur, D.; Yin, X.; Rago, A.; and Toni, F. 2025. Argumentative large language models for explainable and contestable claim verification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 14930–14939.
- Fungwacharakorn, W.; Paulino-Passos, G.; Verheij, B.; and Satoh, K. 2025. On compatibility between situation outcome cases and logical cases. In *Proceedings of the 20th Edition of the International Conference on Artificial Intelligence and Law, ICAIL '25*. New York, NY, USA: Association for Computing Machinery.
- Gould, A., and Toni, F. 2025. Neuro-argumentative learning with case-based reasoning. In *Proceedings of Machine Learning Research*, volume 284.
- Gray, M.; Savelka, J.; Oliver, W.; and Ashley, K. 2023. Automatic identification and empirical analysis of legally relevant factors. In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Law*, 101–110.
- Gray, M.; Savelka, J.; Oliver, W.; and Ashley, K. 2024. Using LLMs to discover legal factors. In *Legal Knowledge and Information Systems*. IOS Press. 60–71.
- Gray, M.; Savelka, J.; Oliver, W.; and Ashley, K. 2025. Generating legal arguments with automatically identified factor magnitudes. In *Proceedings of the 20th Edition of the International Conference on Artificial Intelligence and Law, ICAIL '25*. New York, NY, USA: Association for Computing Machinery.
- Horty, J. F. 2004. The result model of precedent. *Legal Theory* 10(1):19–31.
- Huang, J., and Chang, K. C.-C. 2022. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*.
- Leofante, F.; Ayoobi, H.; Dejl, A.; Freedman, G.; Gorur, D.; Jiang, J.; Paulino-Passos, G.; Rago, A.; Rapberger, A.; Russo, F.; Yin, X.; Zhang, D.; and Toni, F. 2024. Contestable AI Needs Computational Argumentation. In *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning*, 888–896.
- Li, O.; Liu, H.; Chen, C.; and Rudin, C. 2018. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.
- Liga, D.; Markovich, R.; and Yu, L. 2025. Addressing the right to explanation and the right to challenge through hybrid-ai: Symbolic constraints over large language models via prompt engineering. In *Proceedings of the 20th Edition of the International Conference on Artificial Intelligence and Law, ICAIL '25*. New York, NY, USA: Association for Computing Machinery.
- Nguyen, H.-T.; Goebel, R.; Toni, F.; Stathis, K.; and Satoh, K. 2023a. How well do sota legal reasoning models support abductive reasoning? *arXiv preprint arXiv:2304.06912*.
- Nguyen, H. T.; Goebel, R.; Toni, F.; Stathis, K.; and Satoh, K. 2023b. A negation detection assessment of gpts: analysis with the xnot360 dataset. *arXiv preprint arXiv:2306.16638*.
- Paulino-Passos, G., and Toni, F. 2021. Monotonicity and noise-tolerance in case-based reasoning with abstract argumentation. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, 508–518. California, CA, USA: IJCAI Organization.
- Plaat, A.; Wong, A.; Verberne, S.; Broekens, J.; van Stein, N.; and Back, T. 2024. Reasoning with large language models, a survey. *arXiv preprint arXiv:2407.11511*.
- Prakken, H.; Wyner, A.; Bench-Capon, T.; and Atkinson, K. 2015. A formalization of argumentation schemes for legal case-based reasoning in ASPIC+. *Journal of Logic and Computation* 25(5):1141–1166.
- Rotolo, A., and Sartor, G. 2023. Argumentation and explanation in the law. *Frontiers in Artificial Intelligence* 6:1130559.
- Sartor, G. 2002. Teleological arguments and theory-based dialectics. *Artificial Intelligence and Law* 10(1):95–112.
- Sukpanichnant, P.; Rapberger, A.; and Toni, F. 2024. Peer-arg: Argumentative peer review with LLMs. In *Proceedings of the First International Workshop on Next-Generation Language Models for Knowledge Representation and Reasoning (NeLaMKRR 2024)*, NeLaMKRR 2024.
- Toulmin, S. E. 2003. *The uses of argument*. Cambridge university press.
- Čyras, K.; Satoh, K.; and Toni, F. 2016. Abstract argumentation for case-based reasoning. In *Fifteenth International Conference on the Principles of Knowledge Representation and Reasoning*, 243–254. CA, USA: AAAI Press.
- Wang, C.; Deng, Y.; Lyu, Z.; Zeng, L.; He, J.; Yan, S.; and An, B. 2024. Q*: Improving multi-step reasoning for LLMs with deliberative planning. *arXiv preprint arXiv:2406.14283*.

Specific Domain Ontology Construction Using Large Language Models

Vivian Magri Alcadi Soares^{1,2}, Renata Wassermann^{1,2}

¹University of São Paulo (USP)

²Center for Artificial Intelligence (C4AI)

{vivian.soares@alumni.usp.br, renata@ime.usp.br}

Abstract

Ontologies are useful structures to organize and maintain information that can be understood both by humans and systems. However, since their manual crafting is a laborious task, many specific domains lack reference ontologies. The outstanding ability for understanding natural language demonstrated by the Large Language Models (LLMs) has motivated their application to aid on a variety of fields, including on ontology development. This work presents the experimentation with a technique that uses LLMs in the role of domain experts to build conceptual hierarchies for a given initial concept. Twenty ontologies automatically constructed for the domain of the Brazilian maritime territory (a.k.a the Blue Amazon) using GPT-3.5 and GPT-4 were then evaluated by human experts. The models were able to construct overall coherent conceptualizations of the domain, but none of the outputs was completely satisfactory as a representation of the context without refinement.

1 Introduction

Developing an ontology defines a common vocabulary for researchers who need to share information in a domain, and is akin to building machine-interpretable definitions for a set of concepts in a domain and relations among them (Noy and McGuinness 2001). The manual construction of ontologies, however, is a strenuous endeavor, that requires appropriate knowledge about the domain. Many (semi-)automatic ontology extraction methods have been proposed over the years, ranging from basic rule-based and statistical approaches to complicated machine learning and hybrid architectures. Nevertheless, the problem of obtaining a well-structured, relevant and coherent with the desirable application ontology without a fair amount of human labor persists.

In that scenario, approaches using Large Language Models (LLMs) to develop ontologies have shown promising results. The methods seek to leverage the linguistic abilities these models gain on their extensive pre-training, among other techniques that have been proving effective with this technology, to aid on tasks like concept and relation extraction and refinement of ontologies.

Such techniques appear as a promising to bridge the gap of ontologies for specific domains that haven't yet been provided a gold standard. such is the case of the Brazilian maritime territory, a vast area with approximately the same size

as the Amazon rainforest and often referred to as Blue Amazon. It is a region of invaluable importance because of its economical resources, like the oil and gas reserves, its multiple ecosystems, with the rich fauna and flora, the commercial value of the region for Brazil, and even for its key role in climate regulation. Yet, the Blue Amazon is not well-known to the wider public, or even to those living in the coastal region of Brazil. Information about it is dispersed in academic volumes and government reports, or in obscure databases (Pirozelli et al. 2022).

This research propose an experiment with a LLM-leveraged technique to aid on the organization of information about the specific domain of the Blue Amazon. Since the application of such methodologies to ontology learning is still recent and the evaluation process is not standardized. To assess the suitability of a LLM-enabled algorithm to the task, it was considered essential to construct and conduct a thorough assessment of the outputs, involving humans with extensive knowledge of the domain.

The following sections will present an overview of the current stage of ontology development aided by large language models. Next, we present the details of our experiments and the evaluation method build to assess the quality and usefulness of ontologies automatically generated. We proceed with the presentation of the results and the analysis of the outcomes. Finally, we present the main conclusions and the possible directions for future works.

2 Literature Review

Handcrafting big ontologies is a difficult task. Over the past decades, various methods using linguistic, statistical, and logical techniques, usually associated with steps of the process known as the Ontology Learning Layer Cake have contributed to the improvement of automatic ontology development (Asim et al. 2018). More recently, we observed a great rise in the use of deep neural network-based methods for Ontology Learning (OL) procedures (Reshadat et al. 2023) (Amalki, Tatane, and Bouzit 2025). According to (Reshadat et al. 2023), the key advantage that makes Deep Learning (DL) such a powerful approach is that the feature engineering procedure is done automatically. Also, (Du et al. 2024) mention how extensive works have demonstrated that deeper analysis excels in understanding texts compared to shallow learning.

However, deep learning techniques come with some drawbacks. Ontologies require precise representations, and these models show limited success for taxonomy induction (Du et al. 2024). They achieve good performance extracting information on the form of knowledge graphs (KG), but lacking the formal, hierarchical organization of concepts of an ontology that enable further reasoning. They also might struggle with capturing subtle semantic nuances or understanding contextual variations within domain-specific terminology (Du et al. 2024).

(Du et al. 2024) also claims that adapting DL models to new domains and utilizing transfer learning techniques are challenging tasks. Usually these approaches require an annotated corpus with concepts and the relations between them (Reshadat et al. 2023), and acquiring annotated ontologies or extensive labeled data for specific and complex domain knowledge for the training might hinder DL widespread application for OL, not to mention these models large demand for computational resources (Du et al. 2024). On top of that, building and fine-tuning deep learning models require specialized expertise, which can limit accessibility and practical implementation for non-experts when applying to different domain ontologies (Du et al. 2024).

Amidst this landscape, the emergence of Large Language Models (LLMs) stands as a disruptive force (Du et al. 2024). The evolution of the selected publications applying DL for OL in (Amalki, Tatane, and Bouzit 2025) show a spike in 2023, probably due to the emergence of advanced models such as transformers and LLMs, marking a peak in innovation. (Du et al. 2024) claims these models exhibit a remarkable aptitude for understanding semantic nuances, capturing context and inferring relationships among entities, and that their application in ontology learning holds the promise of addressing many of the aforementioned challenges. Leveraging the prowess of pre-trained language representations, few-shot learning techniques, and harnessing the inherent linguistic and conceptual understanding embedded within these models could mitigate the data dependency issue, domain adaptation, and computational resource challenges, enabling ontology construction with smaller datasets and lower computational requirements (Du et al. 2024).

The work of (Amalki, Tatane, and Bouzit 2025) reports that the most common tasks approached in the reviewed publications are construction, enrichment, and population of ontologies. (Du et al. 2024) summarize empirical attempts to verify if LLMs are suitable for ontology learning tasks, emphasizing their focus on term typing, taxonomy building, and non-taxonomic relations. Despite the fact that some publications were presented where LLMs have shown potential to effectively capture the semantic relationships between types and discover non-taxonomic relations, according to (Du et al. 2024), at present, most of the existing researches on ontology learning using large language models focus on hierarchical relations. They also claim several studies have indicated that the utilization of large language models for facilitating the identification of taxonomy significantly mitigates the need for manual intervention.

(Amalki, Tatane, and Bouzit 2025) point out that experimental and empirical approaches in the research analyzed

are prevailing. They view this as revealing primary concern in method development. They also highlight the predominance of biological and life sciences in the application of OL. Together, the domains of bioinformatics, biomedical, and healthcare represent 35.4% of the studies. They argue, however, that the category of other domains (9 domains), with a total of 29.2%, demonstrates not only the breadth but also the versatility of these techniques for ontology learning in capturing the diverse range of niche and emerging fields.

Among the techniques applying LLMs to build ontologies, the work of (Funk et al. 2023), using OpenAI’s GPT 3.5 API to fully automatically construct simple ontologies consisting of hierarchical conceptualizations of a domain showed promising results for its simplicity of replication for a variety of domains. Their prompts are mostly based on a mild form of few-shot learning that take as input a seed concept *C0*, provided by the user, which will define the domain; provide a textual description for it; and explore it by identifying subconcepts, which will be inserted into the hierarchy being constructed. The loop continues for every concept that was placed either until the model can’t find any more new subconcepts, or the stop conditions defined by the hyperparameters, like the maximum exploration depth, are met.

The work of (Perera and Liu 2024) also searched the literature and selected 3 research papers addressing different aspects of OL using LLMs for examination. They highlight the dynamic and progressive nature of Generative AI technology, concluding that LLMs continue to prove to be more efficient and scalable compared to traditional ML and manual methods. Fully automated LLM-based approaches, however, are considered feasible but challenging, necessitating human oversight at the current state of Generative AI. The sentiment is shared by (Du et al. 2024), which considers the full automatic construction for ontology by a system, appealing, but not likely to be possible. Their proposal is to investigate the utilization of interactive methodologies that involve domain experts in the knowledge acquisition process as a solution to improve the interpretive abilities of LLMs, as opposed to solely depending on prompting engineering. These debates reflect the ongoing efforts to strike a balance between automation and accuracy (Perera and Liu 2024).

In the recent studies applying DL techniques to OL, we observe that comprehensively analyzing the quality of the constructed ontology continues to be a problem worthy of study (Du et al. 2024). The research highlights the lack of standardization, especially in the areas of evaluation metrics and benchmarks ((Amalki, Tatane, and Bouzit 2025); (Du et al. 2024)). (Amalki, Tatane, and Bouzit 2025) claim to observe a sharp division in the literature. Whereas some fields rely on established benchmarks, like biomedical and bioinformatics domains — particularly for the tasks of ontology annotation and enrichment —, others require further development for evaluation protocols to be consistently effective (Amalki, Tatane, and Bouzit 2025). Both (Amalki, Tatane, and Bouzit 2025) and (Du et al. 2024) recommend future research focusing on creating more extensive and field-specific benchmarks that can effectively measure the accuracy, relevance, completeness, and practical utility of ontologies generated by LLMs for a broader range of domains.

Precision, recall, and F1 score are typical metrics for performance assessment in the field of ML, and many experiments with LLMs for ontology development employ them to evaluate a model’s ability to correctly identify relevant ontology components while minimizing the inclusion of irrelevant or incorrect content (Perera and Liu 2024). (Amalki, Tatane, and Bouzit 2025) adverts, however, that although these evaluation metrics are important, their central role in evaluating a range of ontology learning tasks probably points to serious limitations of related research work.

Different from a single information extraction task like relation extraction or event extraction, OL is a procedure to construct an ontology, containing other elements and steps. Properly evaluating them is complex. (Du et al. 2024). Some tasks require specialized evaluation measures to capture the distinct aspects of their processes, suggesting a balance between standardization and adaptability in ontology learning evaluation practices (Amalki, Tatane, and Bouzit 2025).

(Perera and Liu 2024) claims recent developments in this field have seen the integration of detailed evaluation metrics and techniques to further enhance the quality and applicability of the generated ontologies. Among the incorporated techniques, they mention user feedback. Many works in the reviews of (Perera and Liu 2024) and (Du et al. 2024) mention human evaluation, either in the form of supervision by the team conducting the experiments, such as in (Funk et al. 2023), or as validation by domain experts (Giglou, D’Souza, and Auer 2023). (Perera and Liu 2024) defends that, by combining computational efficiency with human insights, we ensure the developed ontologies are not only technically sound but also practically meaningful.

3 Methodology

The development of this project involved two main parts, the generation of ontologies related to the context of the Brazilian Ocean with the algorithm described in (Funk et al. 2023) and made available by the authors¹, and the construction and conduction of the evaluation of the outputs by the domain experts. The full outputs of the executions analyzed in this research, as well as the code versions and models of the questionnaires created, may be seen on the repository <https://github.com/Vivian-Magri/Ontology-for-Blue-Amazon/>.

3.1 Ontology Generation

The algorithm used for the ontology generation can be customized by a series of hyperparameter. Generally speaking, they define the limits for each execution, the caching of generated information and the space of probability for the tokens in the response. Two of those parameters were considered of particular interest, Exploration depth, that defines up to what point concepts will be explored, and Frequency threshold, which determines the minimum a concept must appear on the algorithm’s listing to be considered. The first stage of the experimentation was a grid search to determine which values for them seemed more suitable for the generation of ontologies on the desired domain. Using OpenAI’s

GPT API and the model 3.5 turbo, the main model operating by the time of the experiments (March 2024) — which was also the model mainly used on the researchers’ experiments while designing the algorithm —, We tested the following combination of values, in accordance with the authors’ recommendations of range for tests:

- Exploration depth: 2 and 3
- Frequency threshold: range from 5 to 20, with a step of 5

The other hyperparameters were kept on their default value, with the exception of the Prompt caching, that was deactivated, matching Sample caching, deactivated by default, to avoid as much as possible interferences between executions through previously stored information. As initial concept (*C0*), we first attempted to input “Blue Amazon” directly, but the executions of the algorithm finished without obtaining any verified sub-concepts for it. As a workaround, “Brazilian Water Resources”, a more general named concept, but still related to the original theme, was chosen.

The analysis of the produced ontologies showed that, as expected, lower frequency thresholds allow for a broader expansion of concepts, which makes them useful for exploration. Combined with a higher exploration depth, however, the expansion might go too far, and the inclusion of sub-concepts that do not relate well with the context increases. On the other hand, from the frequency threshold of 15 on, we observe a quite restricted expansion, which makes the results seem incomplete. From these observations, we concluded that the combination of Exploration Depth = 3 and Frequency Threshold = 10 produced the most interesting result. These parameter values seem to fit the theme, since it’s neither a widely defined concept, but includes themes that have been hierarchized on subjects such as ecology. Therefore, these were the chosen values for this set of hyperparameters for the following experiments with concepts related to the Blue Amazon.

For the second and main stage of ontologies generation, the chosen test concept was “Coastal ecosystems”. We looked for an initial concept that would be specific enough to avoid overly generic ontologies, while not being too limited to allow for interesting expansions. Also, it is a concept defining a subdomain well under the Blue Amazon domain, but that could occur in different contexts, which allowed the evaluation of this aspect as well.

Considering what was observed on the first tests on how the results would change drastically due to small variations on how the initial concept was input, we decided to test the chosen concept in different ways to compare the results. The aspects tested in combination were:

- first letter of the central concept capitalized or not (keeping the rest in lowercase in both cases, as the concepts normally outputted by the algorithm);
- referencing Brazil or not;
- concept in English or in Brazilian Portuguese (PT-BR);
- request outputs to be in Brazilian Portuguese or not specified (only for the concepts in PT-BR).

¹<https://git.informatik.uni-leipzig.de/hosemann/onto-llm>

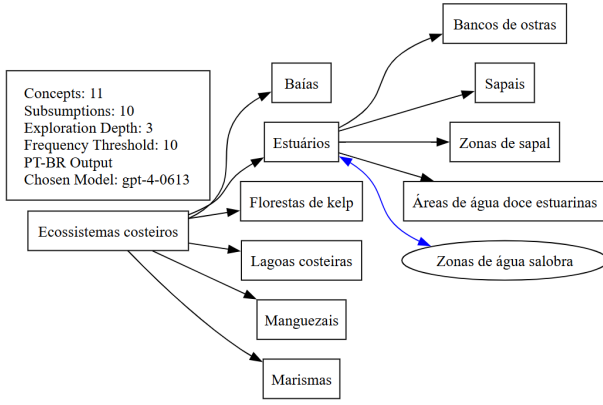


Figure 1: Graphic output generated by passing "Ecosistemas costeiros" as $C0$, 3 as Exploration Depth and 10 as Frequency Threshold to the PT-out code version of the algorithm connected to GPT-4 API

Akin to the choice of including or not a direct reference to Brazil on the input, the variation on the language was introduced to investigate whether using the country's official language would affect the context of the outputted ontologies. At the same time, we thought it would be interesting to see the effect of testing a language the active GPT models already exhibited good ability to communicate with, but which is not the one they were mainly trained in.

For a similar reason, we decided to assess the effect of inhibiting the observed language mixing for $C0$ s in PT-BR by creating a version of the code, that we will refer to as PT-out code version, where we added instructions to the original prompts related to the listing, naming or description of concepts requesting the answer to be given in Brazilian Portuguese. Another small edition to the prompts was made to facilitate executions using other GPT models. The intention was comparing the executions of this stage using GPT-3.5 to their reproduction on the larger and most recent model of the family at the time, GPT-4, to measure if it would improve the perceived quality of the results.

Table 1 summarizes the 26 executions, divided by the model that was used, that constitute the outputs that were considered for the in-depth human evaluation stage. The column $C0$ corresponds to the input passed as initial concept in each run of the algorithm, and the next two columns show how many concepts and subsumptions appear in the output. The last column is the register of the date each execution occurred. The entries listed with the prefix "PT-BR-out" correspond to the executions using PT-out code version. The input "Coastal ecosystems" was tested twice per model to be considered as a parameter of each model's normal variation.

The output of an execution include different formats of representation for the resulting ontology, such as an OWL file, with all concepts and their definitions, and an SVG file, that allows for the visualization of the produced hierarchy. Figure 1 is an examples of the latter. In the image, the first box from top to bottom on the left shows a summary of information about the respective execution, informing how

many concepts were added to the built hierarchy and how many subsumptions connecting them there are. They also inform the chosen values for the hyperparameters Exploration Depth and Frequency Threshold, and if the output was generated by the PT-out code version, in which case this box will contain the headline "PT-BR Output". The rest of the boxes each show concepts placed on the ontology, starting from the concept $C0$, given as input, with the arrows going from a concept to its subconcept. The oval shape connected to a concept by a blue arrow shows how concepts considered synonyms by the model are represented.

3.2 Evaluation

All ontologies produced by the model that turned out with at least 10 concepts were considered valid and had specific forms prepared for each to be filled out by the team of domain specialists. As it can be deduced from table 1, the evaluation phase involved ten forms per model. The questionnaires were formulated mostly based on the works of (Tello and Gómez-Pérez 2004), focused on creating metrics that will allow an objective scaling of the adequacy of an ontology to the desired goals; and of (Almeida 2009), geared towards a more qualitative evaluation. Since the goal is to both perform analysis on the quality and correctness of the output for each execution and also compare all the results, it was considered adequate to gather insight from both types of approaches. (Tello and Gómez-Pérez 2004) motivated the idea of grouping the topics under evaluation hierarchically and the preparation for the calculation of a numeric score for each ontology. Some of the factors they suggest on the evaluation of the content of an ontology, as well as the use of five graduation scales as default for scoring, also served as inspiration. From (Almeida 2009), the main influences were the criteria related to information quality and the insights of how to use the questions to assess how well knowledge on the domain was being transmitted and to what degree an ontology was succeeding in the goal of modeling the real-world concept.

The aspects that were assessed by the specialists are listed below. The ones on the higher-level are called the dimensions. Some of these were defined through a set of factors, that would then be aggregated to compose the score for that dimension. The aggregation of the scores for all the dimensions of an ontology determined their total score. Note that, for clarity, we sometimes refer to concepts positioned on a pair of concept-subconcept as parent-child.

- Accuracy
 - quality of the definition
 - appropriateness of the chosen name
 - how related the concept is to its parent(s)
- Relevance — how much this subconcept adds knowledge by being positioned as a child of its parent(s)
- Coverage — whether the children of this concept are sufficient to explain it
- Precision
 - positioning in the ontology

Model	C0	Number of Concepts	Subsumptions	Date of Execution
3.5 Turbo	Brazilian coastal ecosystems	2	1	15/05/24
	Brazilian Coastal ecosystems	1	0	15/05/24
	coastal ecosystems	5	4	15/05/24
	Coastal ecosystems	18	18	15/05/24
	Coastal ecosystems	20	22	15/05/24
	ecossistemas costeiros	36	49	15/05/24
	Ecosystemas costeiros	24	33	15/05/24
	ecossistemas costeiros brasileiros	33	35	15/05/24
	Ecosystemas costeiros brasileiros	12	14	15/05/24
	PT-BR-out - ecossistemas costeiros	18	22	16/05/24
	PT-BR-out - Ecosystemas costeiros	23	24	16/05/24
	PT-BR-out - ecossistemas costeiros brasileiros	26	35	16/05/24
	PT-BR-out - Ecosystemas costeiros brasileiros	10	12	16/05/24
4	Brazilian coastal ecosystems	19	19	20/08/24
	Brazilian Coastal ecosystems	51	59	14/09/24
	coastal ecosystems	86	145	16/05/24
	Coastal ecosystems	62	72	17/05/24
	Coastal ecosystems	94	133	17/05/24
	ecossistemas costeiros	77	106	20/08/24
	Ecosystemas costeiros	99	141	06/09/24
	ecossistemas costeiros brasileiros	33	40	20/08/24
	Ecosystemas costeiros brasileiros	51	59	14/09/24
	PT-BR-out - ecossistemas costeiros	3	2	05/09/24
	PT-BR-out - Ecosystemas costeiros	11	10	05/09/24
	PT-BR-out - ecossistemas costeiros brasileiros	1	0	05/09/24
	PT-BR-out - Ecosystemas costeiros brasileiros	1	0	05/09/24

Table 1: Summary of the main experiments

- if the concept is indeed a subconcept (or if is closer to being a part, instance, or something else)
- precision of its connections (should penalize both missing and unnecessary connections)
- adequacy of the synonym (if any)
- Information Design
 - depth (number of levels)
 - size (number of concepts)
 - overall evaluation (general grade for the ontology)

All items listed, except for the ones under the dimension Information Design, needed to be analyzed at the concept level, considering each of their connections. Since, to the best of our knowledge, it was the first in-depth evaluation of the algorithm, it was considered important to go into this level of detail to have a parameter over GPT’s suggestions for each element of a hierarchy. The assessed aspects were rated mostly using a scale of 1 to 5, where 1 is the top evaluation, and 5 is the worst. By the time of the formulation of the first questionnaire template, the choice of using an inverted scale seemed more adequate so the other questions would match more closely the Precision assessment of the missing connections that was included, and the pattern was then followed through the whole evaluation phase for coherence. Questions that presented a different scale on the

form had the answers rescaled for the analysis². The forms also provided the opportunity for the respondents to express some impressions that would not be completely translated into a quantitative metric, but were qualitatively analyzed to compose the results.

The team of evaluators was composed of five academics with a background related to the study of the Ocean, from fields such as Geosciences, Environmental Resource Management and Oceanography, most with experience in projects or studies related to Sustainability. The composition had diversity in age, gender, as well as in the educational level, ranging from undergraduate to post-doctor, and including some specialists with experience in lecturing. While answering the questionnaires, they analyzed the constructed hierarchies considering the visual outputs and the definitions the LLMs provided for each concept. Since the suitability of ontologies also depends on the context of their application, it was established that their assessment should be based on what an elementary school student should be taught about the topic defined by the root concept of each given ontology.

Before the actual evaluation start, a test round was conducted, preparing forms for two smaller outputs, which were replied by the specialists after a basic explanation, followed

²For further detail of the conversions, refer to the results processing codes on the repository

by a feedback session to both clarify their doubts and to collect their insights to improve the evaluation. Then we proceeded with the assessment of the ontologies produced with GPT-3.5, where all ten valid ontologies were evaluated by all five evaluators. When the executions using GPT-4 were concluded, however, given the size of the outputs produced, adjustments on the evaluation dynamic proved to be necessary. As a result, simplifications were made on some steps of the forms, yet managing the transition to preserve compatibility with the former template and to avoid losing effectiveness on the evaluation. Also, each specialist evaluated only four GPT-4 outputs. Even after the modifications, most of the forms prepared for this round still required more than double the time of the previous ones to be completed. Therefore, each form was only assigned to two evaluators, taking into consideration the size of the corresponding ontology to keep the load of work of each expert similar.

After the conclusion of the specialists' activities with the prepared forms, we collected the answers, calculated the metrics and analyzed the qualitative results. The results will be presented and analyzed in the next section.

4 Results

As mentioned in the previous section, all the outputs with 10 concepts or more were evaluated by the domain experts, and exactly ten ontologies made with the aid of each of the models achieved the minimal required expansion, meaning they were reviewed and received scores on five dimensions. No *C0* was completely left out of the evaluations, but some only achieved the minimal expansion required on one of the models. We can observe in table 1 that these initial concepts that had insufficient expansion on one of the models often had very significant expansions on the other (considering the average expansion of each model, of course), making the difference in the pattern for the operation of the same algorithm on each model clear.

Table 2 shows the aggregated means for every evaluated dimension, as well as for the mean of the metrics and of the grades directly attributed by the specialists at the end of each assessment, as a general grade for that ontology. It is also convenient to reinforce that, following the logic of the scale, a smaller grade means a better rating. GPT-4 received slightly better general grades, both considering the mean of the calculated metrics (Total) and the mean of Direct Evaluation. The difference, however, is quite small, especially for the latter. Interestingly, for both cases, metrics amount to better scores than direct evaluation.

GPT-4 was better evaluated in most metrics, but GPT-3.5 surpassed it in Precision and in Information Design. In most of them, the difference was not so significant, varying around 0.2 points. It is more sensible, though, in the Coverage dimension, amounting to circa 0.9 point. It seems coherent to assume it is related to the smaller expansion we observe in most ontologies made with GPT-3.5. On the other hand, the greater expansion seems to make some of the outputs of model 4 excessive and less precise.

Figure 2 shows the means of these metrics aggregated by *C0*, and Figure 3 reveals the metrics for the repeated *C0*s separately for an examination of how much they diverged.

The total size of the bars translates to the Total score without the regularization of the average calculation. In the GPT-4 evaluations, all initial concepts had Information Design as the dimension with the biggest bar, meaning their worst score. This is also true for most of GPT-3.5 cases, with two exceptions. In the remaining dimensions, model 3.5 was more consistent in the rankings of the averages in each *C0* than model 4. The ranking order for GPT-3.5 was, from best to worst, Precision, Accuracy, Relevance, Coverage, and Information Design, with few cases of inversion of one position with another in an adjacent rank.

For GPT-4, there was greater variation overall. As patterns we observe that, in addition to Information Design being the worst, for the most part, Precision is the second worst - these were, in fact, the two dimensions in which model 3.5 surpassed model 4 in the overall average. It is also possible to highlight that the Accuracy scores are among the best for most — with the exception of the only execution with coerced output in PT-BR that achieved the minimal required expansion for assessment in GPT-4, which strayed from the pattern in the evaluations in general —. Also, we notice that Relevance is in the central position of the ranking, or adjacent to it, for all *C0*s.

Comparing the ontologies that received an evaluation with their counterparts from the other model, we can see that model 4 outperformed model 3.5 in all totals, and in practically all metric comparisons (apart for Information Design). The exception was in *ecossistemas costeiros*, which is basically an outlier for model 4. It is also noticeable that the number of concepts of model 4 exceeds the equivalent of model 3.5 in almost all but *ecossistemas costeiros brasileiros*, where both models had the same output size, and in those with PT-BR_out. This indicates that size is not the only factor, since, even in these cases, the evaluations favored model 4.

Analyzing the repeated executions separately, we notice in Figure 3 variations between them, especially for GPT-4. In GPT-3, they are very close in number of concepts, and the score variations in each dimension are modest. In model 4, the results show a greater divergence. The second execution generated about 50% more concepts than the first, and variations on the scores are also more evident. For both models, though, coincidentally or not, the first execution was better evaluated. We also noticed that GPT-4 performed better overall than GPT-3.5 in all executions for this *C0*.

Figure 4 presents the relation of the average Total score for each initial concept and the number of concepts included by each model, in the case of Coastal ecosystems, the average of those metrics. We observe that there is no well-defined tendency overall. When we look at model 3.5 alone, the tendency for all of them is to improve the score as the number of concepts increases. Nevertheless, for model 4, an apparent inversion in this tendency occurs past a certain point of the number of concepts.

Looking at the scores for Information Design in Figure 2 we might see a parallel in those tendencies. That seems related to the participant's ratings on aspects Depth and Quantity. The couple of aspects, together with Direct Evaluation, compose the Information Design score, evaluating the on-

Model	Accuracy	Coverage	Precision	Relevance	Information design	Total	Direct Evaluation
3.5	1.784595	2.464992	1.637863	2.044545	2.549630	2.096325	2.848889
4.0	1.566312	1.567132	1.860130	1.870655	2.849074	1.942660	2.822222

Table 2: Average evaluation by metrics and by direct evaluation by model

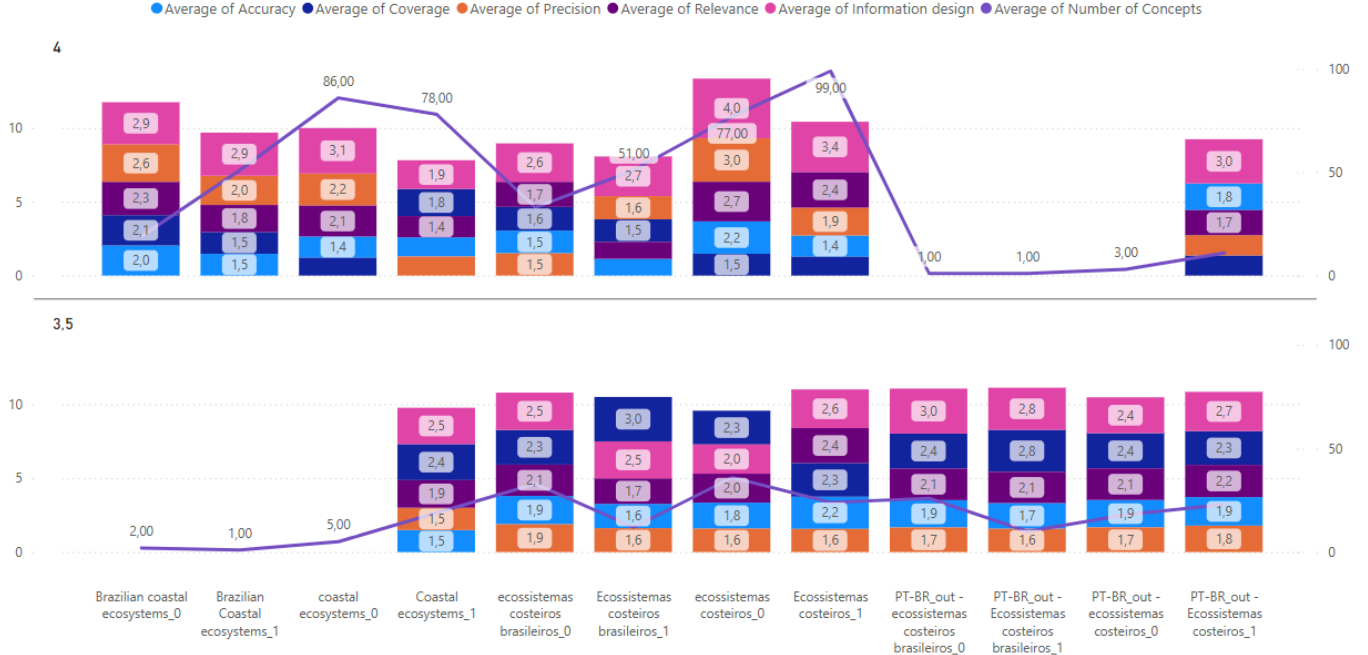


Figure 2: Comparison between the average of the evaluated metrics (bars) and the number of concepts (line) for each model by initial concept

tology as a whole. While GPT-3’s outputs received mostly “is sufficient” and “is moderately insufficient” as answers to how they judged the amount of concepts and levels, respectively, in each hierarchy, GPT-4 received mostly “is quite exaggerated” on both.

Taking a closer look at how the types of variances on the *CO* impact the executions and the corresponding evaluations of the outputs, we observe that capitalization on model 4 produced larger ontologies, on average, with consistently better evaluations. For GPT-3, although in general, capitalized *CO*s also had slightly better expansions, there is almost no difference in the grades.

The language factor seemed to have a greater impact on GPT-3.5. Almost all *CO*s that achieved the 10 concept threshold for evaluation were in Portuguese. The only initial concept in English to have expressive expansion was Coastal ecosystems, with a median number of concepts regarding the results for the model. The expansion of Model 4 is less constant in terms of language. The group that was to be evaluated is composed of half of each type, but the grades were better for *CO*s in English. However, its performance in terms of expansion is much worse when there is an explicit request for output in PT-BR. The two models seem to expand less with the PT-BR_out variation, in fact, but GPT-4

suffers a greater impact.

As for the reference to Brazil in the *CO*s, the concepts were expanded more by model 4 without it. On GPT-3.5, this also occurred, but there was a close difference, and the results are actually a blend. Regarding the scores, the difference between the Totals is greater in model 3.5. The impact of this variation was also examined regarding the mixing of languages in the outputs, which sometimes occurred for *CO*s in Portuguese. For GPT-4, in all executions with initial concepts in PT-BR that had relevant expansion, there was some level of language mixing. The exception was the one generated with the PT-BR_out version of prompts. As expected, in model 3.5, the ones with the idiom coercion also did not have any mixing. On the ones without the restriction, the mixing was less frequent than for GPT-4, with only two outputs exhibiting that behavior.

Analyzing the general comments the evaluators left for the ontologies, the theme of exaggerated expansions for the outputs of model 4 is recurrent. Many reported unnecessary concepts and excessive repetition of concepts within the ontology (usually in the form of multiple inclusions not marked as synonyms of the same concept, with minimal variation. For example, adding to the hierarchy both its plural and single form, or adding versions carrying spe-

cific adjectives that do not bring relevant new information. There are also numerous mentions of incorrect or confusing groupings and classifications, and even the invention of terms. Some also commented on finding some definitions limited to a restricted context, e.g., only covering physical aspects and not approaching the characteristics of life in the ecosystem.

For the model 3.5, some of the same critics appeared on the feedbacks, such as the invention of terms, usually naming incorrectly a concept that does exist; problems in the organization of the hierarchy; lacking descriptions; and some unnecessary concepts or repetition unmarked as synonyms as well — although critics on this topic were less recurrent and emphatic than for GPT-4. In fact, it was more frequent the contrary complaint, recommending the inclusion of terms or a greater expansion in specific points. Missing links were also mentioned, as well as incorrect generalizations and inconsistency in granularity.

After all the evaluation forms had been completed, a voluntary form was passed to the specialists requesting they shared their thoughts on the concept of ontologies, the results, and their participation in this research. Most of the respondents declared that, previous to their engagement on the project, they had little or no knowledge of ontologies. After the experiment, their impression is that this form of organization of knowledge is useful. They see application for it, especially for educational purposes, facilitating the exposition of knowledge in a didactic way.

The majority of the feedbacks considered the outputs from both models, in general, satisfactory. It wasn't unanimous, however. Most of them claim to have found the automation of the construction of the ontologies useful. In one case, though, it was pointed out that, due to the relatively high occurrence of incoherencies, the manual method would be preferable. They all observe the necessity of refinement on the method. The respondents also report problems with the inconsistent level of granularity, with some of the outputted ontologies (or parts of it) being too superficial, while others are too detailed. Plus, it was observed that the deeper the level, the messier the information would appear, demanding more adjustments.

The evaluation of their experience turned out positive, with highlights on the opportunity to reflect on the use of ontologies and on the conceptual structure of the theme, con-

sidering that a good mental exercise. It has been mentioned, however, that the format the evaluation was conducted, with the (long) detailed questionnaires, proved cumbersome at times.

5 Discussion

Examining the assessment of the outputs, we can easily notice that GPT-4 produced, on average, larger ontologies than GPT-3.5, and the metrics were more favorable to the newer and larger model. However, the difference is not so significant, nor is it a consensus, meaning that the considerable difference between the costs is not directly translated to results. Also, the extra expansion included a high level of redundancy and hallucinations. We must observe, however, that the algorithm in the original study had only been designed considering GPT-3, and no variation of language or format on the input had been consistently tested. Maybe prompts for a verbose model, such as the version of GPT-4 tested by the present work, need some refinement to better restrain concept expansion, and to deal with language variations.

The test with the repeated inputs, conducted minutes away from each other in each model, serves as a thermometer for the variability within the models. Analyzing the outputs directly, we observe that, even in the case of model 3.5, where the metrics turned out very similar, the structure has variations. But the variability on model 4 was certainly more pronounced. This kind of inconsistency in a system might raise insecurity for the users. Of course, the temperature of the models can be adjusted as a means to reduce this effect. On the other hand, the outputs would then lose some of the “creative” effect, which is desirable in many applications where LLMs are being employed, and that might be the case for concepts and relations suggestions for an ontology. Thus, finding the balance between “spontaneous” and reliable is another challenge.

The variation in the performance of each model due to linguistic aspects brought some surprises. The fact that GPT-3.5 had satisfactory outputs for *COs* in Portuguese evokes our theory that the availability of content about the Brazilian coast on the internet was more abundant. However, we did not anticipate such difficulty for the expansion of most of the concepts in English. It was expected that the context approached by each could exhibit differences, but we believed the average of concepts would be, if not higher, similar to the inputs in Portuguese. The struggle of the newer model with the coercion for the outputs to be in PT-BR was even more unforeseen, especially considering how GPT-4 expanded well the concepts in Portuguese. It is true that these outputs mixed the languages. But it remains unclear why the restriction not only reduced the amount of expansion but basically extinguished it, and why this effect did not generate the same level of disturbance on model 3.5. It is convenient to mention that these executions for model 4 were each tested more than once, even on different days, to confirm it was not a bug. The results section only presented the data of the original executions since the retrials did not present a significant difference.

It is not possible to define an optimal point for the number of concepts based on our evaluations. More tests of

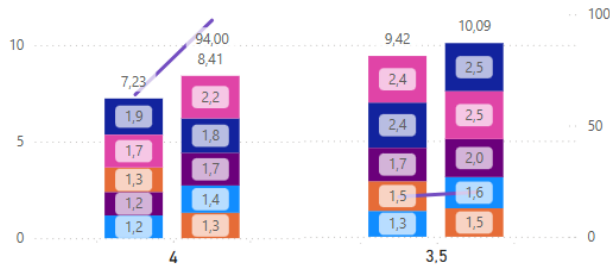


Figure 3: Comparison between executions of the repeated *CO* Coastal ecosystems. The caption and metrics follow as in Figure 2

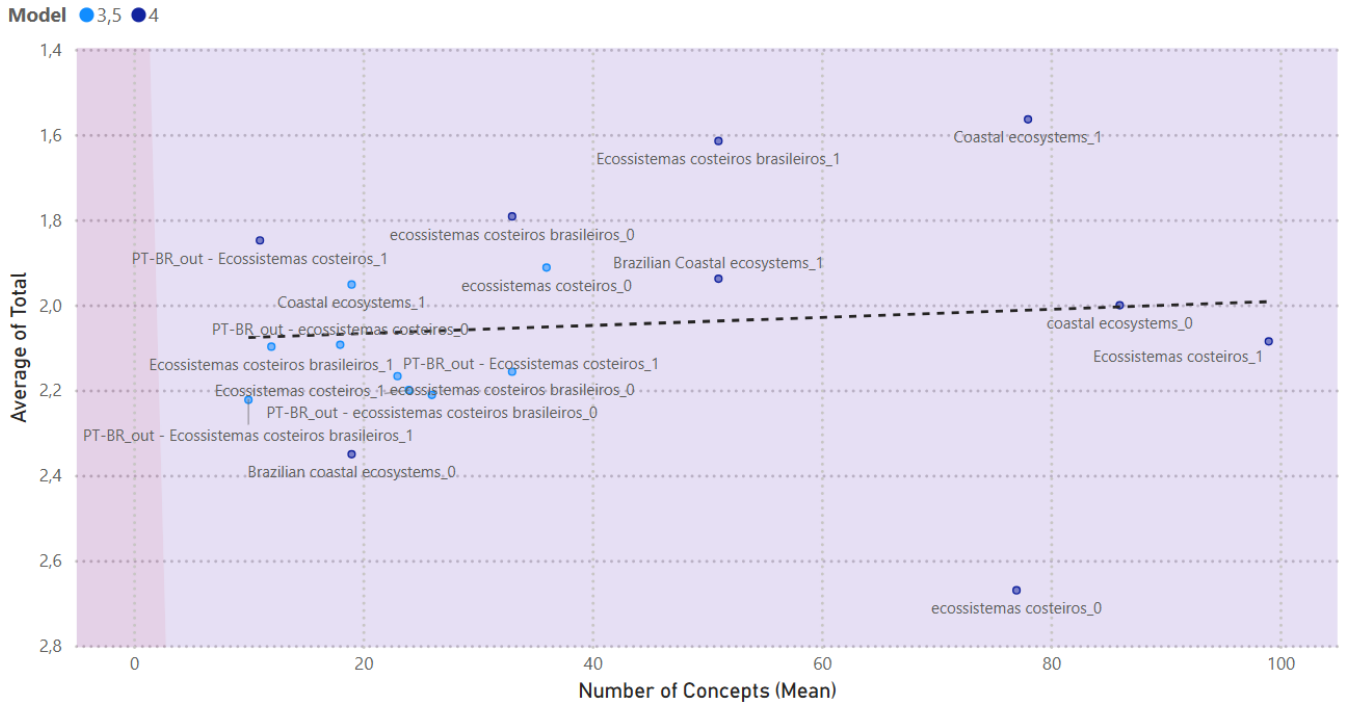


Figure 4: Correlation between number of concepts in the outputs and the average evaluation score (Total) by initial concept. Colors distinguish between the GPT models used on the execution

each type would be needed to draw more accurate conclusions. However, human evaluation is quite costly, so we did not have the conditions to extend the tests. Besides, this kind of evaluation surely is sensitive to differences in participants' opinions due to personal experiences, preferences and knowledge, among other variations beyond our control. Therefore, we understand that the number of tests and evaluators employed in this research is not enough for definitive verdicts.

Still, we judge that none of the models showed the capacity to construct entirely satisfactory ontologies autonomously with this technique. As highlighted, even GPT-4, which achieved the best grades overall, received a great deal of critique on its outputs, and presented a fair amount of hallucinations and inconsistency over the executions. This understanding is, in fact, in accordance with the conclusions and recommendations in the reviewed literature. (Du et al. 2024) states that "while the idea of fully automatic ontology construction is appealing, especially for handling large volumes of data or complex domains, it is worth mentioning that full automatic construction for ontology by a system is still a significant challenge and it is not likely to be possible". And the recommendation to integrate "human-in-the-loop approaches with expert involvement [to] enhance ontology relevance and accuracy" (Giglou, D'Souza, and Auer 2023) is also recurrent, even appearing on (Funk et al. 2023), the base work for these experiments.

Finally, despite some reported confusions, inventions, and problems with the granularity, we consider the models were both successful in extracting concepts pertinent to the do-

main and arranging the hierarchical relations mostly with coherence. The merit is also majorly on the algorithm constructed and the engineering of the prompts conducted by (Funk et al. 2023), not simply on the use of LLMS. The model is the judge, but the systematic search for new terms, the verification process design, and the traversal algorithm for inserting the terms in the graph under construction conducted the models through the steps to complete the task.

6 Conclusions and Future Work

This project could successfully conduct the experiments with the method developed by (Funk et al. 2023) to study the construction of ontologies in the context of the Brazilian Maritime Coast. It was not possible to arrive at a fully satisfactory hierarchical conceptualization to organize the knowledge of the Blue Amazon with the employed methodology. One of the reasons was the tests showed it was convenient to choose a more restrict initial concept than initially planned in order to achieve more substantial outputs for the assessment. The choice of Coastal ecosystems produced diverse results, with the variance on the format of the input to analyze the effect on the algorithm.

Another highlight was the possibility of repeating the experiments with two models of the same family with different sizes, and very different costs, to compare their performance. Additionally, the use of human specialists on the domain allowed for broader examination of the outputs. The evaluation revealed the model GPT-4 performs slightly better overall. Yet, it struggles with drawbacks such as hallucinations and redundancy. Plus, in both models there was

inconstancy on the level of granularity of the conceptualization and incoherencies on some relations. Thus, the produced ontologies should not be directly utilized without a revision. They may serve, though, as drafts that could be refined and expanded to produce adequate symbolic representation of the domain.

The results serve to corroborate with the predominant view presented on the researched literature that LLMs are suitable tools to aid on the development of ontologies, however, at this point, even advanced models do not demonstrate to be safe to produce functional ontologies on their own. Beyond the technical challenges of the technology, the process of constructing an ontology is entangled with specific motivations. The complete context of the desirable applications, so far, seems to be still too difficult to infuse in AI technology.

Thus, we conclude the method is valid and particularly interesting to aid in the expansion of OL to domains where existing specific ontologies are lacking, especially if applied under the supervision of domain experts.

We believe this research also has great potential for expansion beyond the scope of this project. As suggested by the article (Funk et al. 2023) itself, as a relevant direction for future work, it's also possible to experiment with the construction of ontologies that are more expressive, adding other kinds of relations and, possibly, even rules, as disjointness, for example. Another interesting suggestion they make is fine-tuning for domain-specific ontology construction, training the model with curated information about the intended subjects.

Another line of work that could be of great value is the modification to the prompts to include the human on the loop approach. Note that, as described by (Noy and McGuinness 2001), ontology development is necessarily an iterative process. After an initial version of the ontology is defined, it can be evaluated and debugged by using it in applications or problem-solving methods, or by discussing it with experts in the field, or both. They state that, as a result, the original design will almost certainly need to be revised. Therefore, turning this technique into a practical instrument for specialists to work with ontologies seems like a natural idea.

7 Acknowledgments

This work was carried out at the Center for Artificial Intelligence (C4AI-USP), with the support of the University of São Paulo, the São Paulo Research Foundation (FAPESP, grant #2019/07665-4) and by the IBM Corporation. Vivian Magri A. Soares was also supported by CAPES.

References

- Almeida, M. B. 2009. A proposal to evaluate ontology content. *Appl. Ontology* 4(3-4):245–265.
- Amalki, A.; Tatane, K.; and Bouzit, A. 2025. Deep learning-driven ontology learning: A systematic mapping study. *Engineering, Technology & Applied Science Research* 15(1):20085–20094.
- Asim, M. N.; Wasim, M.; Khan, M. U. G.; Mahmood, W.; and Abbasi, H. M. 2018. A survey of ontology learning techniques and applications. *Database J. Biol. Databases Curation* 2018:bay101.
- Du, R.; An, H.; Wang, K.; and Liu, W. 2024. A short review for ontology learning: Stride to large language models trend.
- Funk, M.; Hosemann, S.; Jung, J. C.; and Lutz, C. 2023. Towards ontology construction with language models. *CoRR* abs/2309.09898.
- Giglou, H. B.; D'Souza, J.; and Auer, S. 2023. Llm4ol: Large language models for ontology learning.
- Noy, N., and McGuinness, D. 2001. Ontology development 101: A guide to creating your first ontology. *Knowledge Systems Laboratory* 32.
- Perera, O., and Liu, J. 2024. Exploring large language models for ontology learning.
- Pirozelli, P.; Castro, A. B. R.; de Oliveira, A. L. C.; Oliveira, A. S.; Cação, F. N.; Silveira, I. C.; Campos, J. G. M.; Motheo, L. C.; Figueiredo, L. F.; Pellicer, L. F. A. O.; José, M. A.; José, M. M.; de M. Ligabue, P.; Grava, R. S.; Tavares, R. M.; Matos, V. B.; Sym, Y. V.; Costa, A. H. R.; Brandão, A. A. F.; Mauá, D. D.; Cozman, F. G.; and Peres, S. M. 2022. The blue amazon brain (BLAB): A modular architecture of services about the brazilian maritime territory. *CoRR* abs/2209.07928.
- Reshadat, V.; Akcay, A.; Zervanou, K.; Zhang, Y.; and de Jong, E. 2023. SCORE: special cargo relation extraction using representation learning. *Neural Comput. Appl.* 35(25):18783–18801.
- Tello, A. L., and Gómez-Pérez, A. 2004. ONTOMETRIC: A method to choose the appropriate ontology. *J. Database Manag.* 15(2):1–18.

Towards a Common Framework for Autoformalization

Agnieszka Mensfelt*, David Tena Cucala*, Santiago Franco*,
Angeliki Koutsoukou-Argyraiki*,[†] Vince Trencsenyi*, Kostas Stathis*

*Department of Computer Science, Royal Holloway, University of London,
Egham Hill, Egham, TW200EX, UK

[†]Department of Computer Science and Technology, University of Cambridge,
William Gates Building, 15 JJ Thomson Avenue, Cambridge, CB3 0FD, UK

{agnieszka.mensfelt, david.tenacucala, santiago.franco, angeliki.koutsoukouargyraiki, vince.trencsenyi,
kostas.stathis}@rhul.ac.uk

Abstract

Autoformalization has emerged as a term referring to the automation of formalization—specifically, the formalization of mathematics using interactive theorem provers (proof assistants). Its rapid development has been driven by progress in deep learning, especially large language models (LLMs). More recently, the term has expanded beyond mathematics to describe the broader task of translating informal input into formal logical representations. At the same time, a growing body of research explores using LLMs to translate informal language into formal representations for reasoning, planning, and knowledge representation—often without explicitly referring to this process as autoformalization. As a result, despite addressing similar tasks, the largely independent development of these research areas has limited opportunities for shared methodologies, benchmarks, and theoretical frameworks that could accelerate progress. The goal of this paper is to review—explicit or implicit—instances of what can be considered autoformalization and to propose a unified framework, encouraging cross-pollination between different fields to advance the development of next generation AI systems.

1 Introduction

Drawing on the long tradition of formalizing human reasoning through mathematics and logic, AI pioneers (McCarthy 1960; Kowalski 1974; Newell and Simon 1976) incorporated ideas from formal systems into functional and logical programming languages for formalizing knowledge and solving practical problems. Building on these foundations, researchers in natural language processing (Weizenbaum 1966) and language understanding (Winograd 1972) began applying formal representations and inference techniques to human language (Warren and Pereira 1982). This enabled computers to reason about the meaning of sentences, answer questions, and interact with users in more human-like ways, thus bridging the gap between symbolic AI and language understanding. These efforts opened the way in the development of semantic parsing (Kamath and Das 2018), a field dedicated to translating natural language into formal meaning representations.

The term *Autoformalization* was originally introduced to describe the automatic formalization of mathematics with interactive theorem provers (proof assistants) such as Is-

abelle/HOL, Lean, Rocq, and Mizar, a field that has grown rapidly in recent years; there is increasing interest in employing deep learning to automate the task of formalization (Wang, Kaliszyk, and Urban 2018; Szegedy 2020; Wu et al. 2022; Jiang, Li, and Jamnik 2024). Earlier, pioneering work employing techniques from machine learning had introduced the automation of the task of formalization referring to it as “corpus-based translation between informal and formal mathematics” (Kaliszyk et al. 2014) and “parsing between aligned corpora”, exploring the challenge: *Is it possible to automatically parse informal mathematical texts into formal ones and formally verify them?* (Kaliszyk, Urban, and Vyskočil 2015). Autoformalization is closely related to semantic parsing (Kaliszyk, Urban, and Vyskočil 2017) and can be seen as an instance of semantic parsing. The distinctive feature of autoformalization would be that—unlike semantic parsing which encompasses a broad range of target formalisms including general purpose programming languages—it is restricted to target formal languages used in logical inference and automated reasoning. Independently of this distinction, with the development of deep learning approaches to natural language processing—particularly the emergence of transformer architectures—the task of translating informal language into formal representations is most often performed using large language models (LLMs)¹. This approach is more general and flexible, and less tied to a specific target language or domain than traditional semantic parsing methods; however, it comes at the cost of introducing new challenges in verifying their less restricted outputs.

Despite its growing use, current approaches to autoformalization remain largely ad hoc, and there is no—outside the original sense of formalizing mathematics—widely agreed-upon definition of what the term encompasses. This lack of clarity hinders communication across subfields of formal methods, proof engineering, and AI. Different communities may use the term to refer to subtly different tasks, or not explicitly consider their work to constitute autoformalization, making it difficult to compare

¹The commonly used models include Claude, Codex, DeepSeek, GPT3.5, GPT4, Llemma, Minerva, Mistral, and Mixtral.

methods, evaluate results, and transfer knowledge. Establishing a clear, comprehensive, and widely accepted definition would facilitate interdisciplinary collaboration and accelerate progress by enabling more systematic benchmarking, evaluation, and knowledge sharing. To address these issues, we propose a common framework for understanding autoformalization.

An additional motivation is the potential role of autoformalization in advancing the next generation of AI systems. Since the emergence of LLMs, there has been growing interest in using them as general-purpose reasoners. However, despite their impressive versatility in natural language processing, LLMs—trained primarily as next-token predictors—hallucinate, producing outputs that are logically inconsistent or factually incorrect. In response, “reasoning” models have been developed to address these shortcomings by incorporating techniques such as chain-of-thought prompting and reinforcement learning. While these models show improved performance on certain benchmarks, they still fall short in reliably performing complex logical inference (Shojaee et al. 2025) and remain costly to train. Furthermore, reasoning in these models is often difficult or impossible to verify formally, raising concerns about their use in high-stakes applications (Nguyen et al. 2025). In contrast, autoformalization leverages a task that large language models naturally excel at—translation—by mapping informal language into formal representations. This offers a potentially more efficient and interpretable alternative, or at least a complementary approach, to building general-purpose LLM-based reasoners. By streamlining the interaction between language models and formal reasoning tools, autoformalization could play a key role in enhancing robustness and accuracy in domains such as mathematics, software verification, legal reasoning, and scientific discovery—where logical reasoning is essential.

2 Review of Existing Usage and Definitions

To propose a unified cross-disciplinary framework, we surveyed 81 research papers focused on the automatic translation of natural language into formal languages used for logical inference and automated reasoning. While we aimed to cover all relevant work in this area, we do not claim that the review is exhaustive. Our objective was not a detailed review of the included papers, but rather to examine both formal and informal definitions of *autoformalization* and to identify tasks that can be interpreted as instances of autoformalization, even if the term itself was not explicitly employed.

We included papers on *autoformalization* as the term has been used since 2018, i.e., referring to the automatic translation of informal mathematics into “formal and verifiable mathematical language” (Wang, Kaliszyk, and Urban 2018). Furthermore, noting that the term *autoformalization* has begun to gain traction outside the context of mathematical formalization (Pan et al. 2023), we also included work on the translation of natural language into other languages that support formal verification, reasoning, or knowledge representation—such as Prolog, PDDL, or OWL—even if these are not languages of interactive theorem provers.

We categorized the reviewed papers into four broad subfields: formalization of mathematics using interactive theorem provers, logical inference and declarative programming, planning, and knowledge representation. These subfields were selected to reflect distinct research communities, each of which may conceptualize autoformalization differently and may have limited interaction or knowledge exchange with others. Nonetheless, the boundaries between these subfields are not always clear-cut. On the one hand, there is sometimes overlap between the categories (e.g., logical reasoning can be used for knowledge representation); on the other hand, additional categories may be warranted as the field evolves.

Formalization of Mathematics with Interactive Theorem Provers

The term *autoformalization* originated in the context of formalizing mathematics in 2018 (Wang, Kaliszyk, and Urban 2018), and as a result, it is widely adopted within this domain. Since the recent emergence of LLMs, users of proof assistants have been experimenting with using LLMs to assist with the formalization of mathematics, yielding mixed but potentially promising results (Karatarakis 2024). All 32 papers reviewed for a definition from this subfield use the term *autoformalization*. Most of works provide only informal definitions of autoformalization, which may reflect the fact that formalization of mathematics is intuitively well understood within the community and does not require defining. These informal definitions typically describe autoformalization as the task of automatically translating mathematical content from informal language into formal languages (Wu et al. 2022; Jiang et al. 2022; Agrawal et al. 2022; Li et al. 2024; Zhou et al. 2024; Ying et al. 2024; Yang et al. 2024; Liu et al. 2025a; Lu et al. 2024; Poiroux et al. 2024b; Chan et al. 2025; Jiang 2025; Azerbayev et al. 2023; Wang et al. 2020; Yu et al. 2025; Poiroux et al. 2024a; Hu et al. 2025b; Wang et al. 2025a). Some authors also explicitly include machine-verifiability of the output as part of the definition (Szegegy 2020; Murphy et al. 2024; Liu et al. 2024b; Liu et al. 2025c; Patel, Saha, and Flanigan 2023; Cunningham, Bunescu, and Juedes 2023; Liu et al. 2025b; Weng et al. 2025; Lu et al. 2025; Jiang, Li, and Jamnik 2024; Gadgil et al. 2022). Only one group of authors attempted a formal definition (Zhang, Quan, and Freitas 2024; Zhang, Valentino, and Freitas 2025).

Logical Inference and Declarative Programming

In this category, we reviewed 30 papers. The target formalisms include first order logic (Pan et al. 2023; Lalwani et al. 2024; Han et al. 2022; Hahn et al. 2022; Yang, Ishay, and Lee 2023; Olsson et al. 2023; Chaturvedi and Asher 2024; Thatikonda et al. 2024; Ryu et al. 2024; Quan et al. 2024; Brunello et al. 2025; Liu 2025; Lee et al. 2025); declarative programming language, Prolog (Mensfelt, Stathis, and Trencsenyi 2024; Mensfelt, Stathis, and Trencsenyi 2025; Borazjanizadeh and Piantadosi 2024) and answer set programming (ASP) (Li et al. 2025; Coppolillo et al. 2024; Borroto Santana, Kareem, and Ricca 2024; Ishay, Yang, and

Lee 2023). Another main line of work focuses on translating problems from natural language into temporal logics such as LTL (Pnueli 1977) and CTL (Clarke, Emerson, and Sistla 1986). These languages extend classical logic with temporal operators to reason about properties that hold over time, with LTL describing linear sequences of states and CTL describing branching time structures where multiple futures are possible from each state. There exist a significant number of recent works focused on translating natural language descriptions to these languages (Chen et al. 2023; Pan, Chou, and Berenson 2023; William et al. 2024; Liu et al. 2024a; Mavrogiannis, Mavrogiannis, and Aloimonos 2024; Xu, Feng, and Miao 2024; Li and Tian 2025; Soroco et al. 2025; Wang et al. 2025c; Wang et al. 2025b); one the reasons of popularity of temporal logics is its relevance in industrial applications.

Only 8 of reviewed papers explicitly used the term *autoformalization*, and all of them defined it informally—typically as the translation of informal or natural language into formal representations (Pan et al. 2023; Lalwani et al. 2024; Mensfelt, Stathis, and Trencsenyi 2024; Mensfelt, Stathis, and Trencsenyi 2025; Soroco et al. 2025; Olausson et al. 2023; Lee et al. 2025; Quan et al. 2024). Among the remaining works, one paper formally defines the task (Liu et al. 2024a), while the rest describe it informally as translation from natural language into a specific formalism. Notably, even though these papers do not use the term *autoformalization*, their definitions are consistent with those that do.

Planning The Planning Domain Definition Language (PDDL) is the de facto standard formalism within the AI planning community (Ghalab et al. 1998), encompassing various extensions ranging from basic STRIPS to first-order, probabilistic, and temporal logics (Fox and Long 2003).

A PDDL planning instance consists of a domain file and a problem file. The domain file specifies predicates (facts) and actions applicable across the entire domain. Actions are defined through preconditions and effects, encapsulating a labeled transition system that alters the truth values of a set of facts. The problem file describes the particular instance, specifying the initial state and the goal conditions.

PDDL formally characterizes logical planning problems where the task is to generate a sequence of actions (a *plan*) to transition from the initial to a goal state. Importantly, PDDL is purely descriptive; it does not inherently provide solutions. Therefore, planners like (Helmert 2006) are required to compute plans. This separation allows LLMs to effectively tackle complex combinatorial problems by translating them into PDDL representations, leveraging specialized logic solvers and thus combining the strengths of LLMs (natural language translation) and automated planning systems (combinatorial reasoning).

In the planning category, we included 16 papers. None of them use the term “autoformalization”. One of the works provides a formal definition of the task (Hu et al. 2025a) as defining a mapping function generating a world model from a natural language description. The rest of

the works define the task informally (Oates et al. 2024; Liu et al. 2023; Aghzal et al. 2025; Smirnov et al. 2024; Xie et al. 2023; Mahdavi et al. 2024; Huang and Zhang 2024; Hu et al. 2025a; Lin et al. 2023; Oswald et al. 2024; Silver et al. 2024; Guan et al. 2023; Sikes et al. 2025; de la Rosa et al. 2024). In some cases the task may encompass generating both domain and problem files (Smirnov et al. 2024), while in others only PDDL snippets are being instantiated, e.g., goals in (Xie et al. 2023). A related but separate line of research involves using LLMs as planners themselves, generating plans from PDDL inputs. This direction falls outside the scope of the present work.

Knowledge Representation Several works that we surveyed (Mateiu and Groza 2023; Doumanas et al. 2024; Caufield et al. 2024; Saeedizade and Blomqvist 2024) focus on translating natural language descriptions of real-world domains into ontologies using the Web Ontology Language (OWL) (OWL Working Group 2012), a formal language used to define and share structured, machine-readable knowledge in terms of individuals, concepts, and relationships between them. Some other works produce ontologies expressed in RDF or RDFS (Brickley and Guha 2014). RDF allows us to express Knowledge Graphs, where nodes represent named entities, node labels represent their individual properties (called “types” or “classes”), and labelled edges represent relations between them. RDFS enriches RDF with information about subsumption, domain, and range relations between classes. Both RDF and RDFS can be seen as subsets of OWL. Works such as (Eells et al. 2024; Abolhasani and Pan 2024) translate natural language to RDFS and RDF ontologies, respectively. Others express ontologies in ad-hoc languages (Tang et al. 2023; Lippolis et al. 2025), which seem to be expressible in OWL or RDFS. In all these works, syntactic validity is checked automatically, and semantic is checked manually, or by running certain conjunctive queries on the produced ontology. None of the reviewed works in this category employs the term *autoformalization* to denote the translation task.

Summary As observed, definitions across domains exhibit notable similarities, even when the task is not explicitly labelled as *autoformalization*. At the core, these tasks involve the automatic translation of natural language into a target language that supports logical inference and automated reasoning. Building on these shared foundations, we now propose a unified conceptual framework that can encompass this broad class of tasks.

3 Proposed Definition

To define autoformalization, we begin by first establishing a definition of formalization. Then, we demonstrate how selected case studies—simple examples from various domains—align with this definition.

3.1 Definition of Autoformalization

We introduce a general definition intended to capture the essential components of autoformalization across a broad

range of domains and settings, while abstracting away from implementation-specific or domain-specific details.

Definition 1. Formalization from informal language L_i to formal reasoning language L_f with respect to a semantic equivalence criterion E is the transformation of an expression in a domain-specific subset of L_i into a well-formed and valid expression in L_f that is semantically equivalent according to E . Autoformalization is formalization performed automatically by a computing system.

Our definition involves four parameters: the informal language L_i , a domain-specific subset of L_i , the formal language L_f , and the semantic equivalence criterion E . These are treated as primitives without formal definition. However, we proceed to clarify their meanings, provide examples, and explain our rationale for including them in the definition.

Informal language An informal language L_i is a collection of meaningful expressions, such as the set of all grammatically well-formed and semantically coherent texts in English. This abstraction deliberately ignores more fine-grained aspects, such as lexical categories or compositional syntax. Membership in informal languages is generally neither decidable nor computable, and their boundaries may be vague or context-dependent. Sometimes we consider *semi-formal* languages: informal languages (like plain English) mixed with elements of a formal language. For example, the mathematical language of proofs is typically semi-formal.

Our definitions refers to a domain-specific subset of L_i because, in a given autoformalization setting, we typically do not seek to formalize all possible expressions in L_i , but only a specific subset of interest that share a common conceptual framework or subject matter. For example, within plain English, we may consider: descriptions of two-player games, specifications of procedural planning settings, and natural language descriptions of real-world systems in terms of their objects, properties, and relations (i.e. informal ontologies), among many other examples.

Formal reasoning language The formal reasoning language L_f is an enumerable set of expressions typically specified by a grammar or formation rules, and is accompanied by a formal semantics that assigns precise and unambiguous meaning to each well-formed expression. It is also associated with a reasoning apparatus—usually a set of inference or transformation rules—that enables the derivation of new expressions from existing ones while preserving semantic properties. Examples of L_f include:

- languages of proof assistants, i.e., interactive theorem provers (e.g., Lean, Isabelle, Mizar, Rocq);
- fully formal logics (e.g., propositional logic, FOL, LTL);
- logic programming and declarative languages (e.g., Answer Set Programming, Prolog);
- planning and CSP languages (e.g., PDDL, MiniZinc)
- knowledge representation languages and formalisms (e.g., OWL, RDF, Situation Calculus, Event Calculus).

The choice of formal language L_f in a given autoformalization setting is guided by two considerations: first, it must be capable of representing all relevant information in the domain-specific subset of L_i ; second, it must support reasoning in a way that allows properties of interest to be verified, both in terms of computational complexity and availability of appropriate verification tools. For example, if our goal is to formalise descriptions of planning scenarios, L_f might be a planning modeling language such as STRIPS.

Semantic equivalence criterion The semantic equivalence criterion E specifies what it means for the obtained formal expression from L_f to preserve the intended meaning of the informal input from L_i . While the goal is for both expressions to ‘mean the same thing,’ formal languages are usually less expressive than informal languages and cannot capture every aspect of the original meaning. Hence, in practice, E is designed to ensure that the formalization preserves the aspects of meaning that are most relevant for the task at hand. For example, in an interactive theorem prover, E may require that the formalization contains sufficient detail to ensure that formalisations of correct proofs (expressed informally in L_i) can be verified by a given tool or reasoning algorithm.

Specifying the semantic equivalence criterion can be challenging, as it requires defining conditions related to the meaning of informal language, which is inherently ambiguous. To address this, we introduce a related concept, the *validation criterion* V , which serves as a computable approximation of E . This distinction is important: E defines what it means for an informal and formal expression to “mean the same thing” as a formal expression, whereas V provides a practical means of verifying whether a given pair of informal and formal expressions satisfies this equivalence.

We conclude this section with several observations that clarify and extend the scope of the definition.

Remark 1. An informal expression in L_i may be formalized in two or more distinct ways into L_f which may differ in their choice of names, syntactic structure, or level of abstraction. Conversely, two distinct informal expressions may be formalized into the same expression in L_f ; this may happen, for example, if they describe the same state of affairs according to the semantic equivalence criterion E , reflecting the inherent variability and redundancy of natural language.

Remark 2. Systems for autoformalization need not be deterministic: the same informal input may be mapped to different formal outputs in different runs. These outputs may or may not be equivalent to one another, but each must satisfy the equivalence criterion E with respect to the input. In practice, it is often desirable for a system to prefer outputs that are syntactically or structurally minimal, promoting conciseness and efficiency.

Remark 3. Although our definition is concerned with equivalence between complete informal and formal expressions, many applications require equivalence to hold at a finer granularity. For example, a multi-sentence informal description of a mathematical argument may be formalized as a sequence of formal statements, each corresponding to a spe-

cific sentence or sub-argument. Such fine-grained correspondence should be subsumed within the semantic equivalence criterion E .

3.2 Case Studies

To further illustrate the flexibility of our definition and its applicability across a range of settings, we next show how it can be applied to capture existing examples in the literature. The case study analysis will focus on what constitutes informal input, formal target language, and how semantic equivalence is evaluated, while abstracting away from implementation details such as the application of in-context learning and self-debugging feedback loops.

Formalization of Mathematics with Interactive Theorem Provers The following example comes from a seminal paper on autoformalization (Wu et al. 2022). In this example, the informal language L_i is the natural mathematical language (plain English extended with mathematical symbols) and the relevant subset can be considered as a chosen set of mathematical problems (the example at hand coming from the 1987 International Mathematical Olympiad), with the target language L_f in this case being that of Isabelle/HOL (Nipkow, Wenzel, and Paulson 2002). Isabelle/HOL is the most widely used instance of Isabelle, an interactive theorem prover for formal verification and proof construction. Isabelle/HOL provides a higher-order logic theorem proving environment. In this context, L_f is Isabelle’s structured proof language *Isar* that includes symbols such as keywords, mathematical symbols, and logical operators, and has a formally defined syntax. The reasoning apparatus is realised by various automatic tools incorporated in Isabelle/HOL (and by a number of external automatic theorem provers and satisfiability-modulo-theories solvers which can be called by the Sledgehammer tool featured by Isabelle/HOL (Blanchette et al. 2025)).

In our example (Wu et al. 2022), the input expression to be formalized in L_i was as follows:

Prove that there is no function f from the set of non-negative integers into itself such that $f(f(n)) = n + 1987$ for every n .

The corresponding formalization in Isabelle/HOL produced by the Codex LLM tool was as follows:

```
theorem
  fixes f :: "nat \ $\rightarrow$  nat"
  assumes "\forall n. f (f n) = n + 1987"
  shows False
```

The statement `shows False` indicates that the assumption of the described function’s existence leads to a contradiction.

The semantic equivalence criterion E in this case requires that all important mathematical details from the original theorem statement are captured in the formal statement. This is crucial because when we later formalize a proof and verify its validity, all essential claims that the proof must establish need to be correctly represented in the theorem statement.

Two validation criteria V were used here to evaluate semantic correctness. First, BLEU scores (Papineni et al.

2002) were used, which measure surface-level similarity between the autoformalized statement and the ground truth (i.e. a manual formalization deemed correct by human experts), but not direct semantic equivalence. The second validation criterion was manual evaluation of selected examples. While manual evaluation reliably assesses semantic equivalence, it is costly and does not scale well. This highlights a fundamental challenge in autoformalization: automating the evaluation of semantic correspondence.

First-Order Logic In (Pan et al. 2023) authors explicitly state that their goal is to extend autoformalization into logical reasoning. In the example from the paper, their relevant subset of natural language (L_i) consists of questions about real-world objects and shapes. The target formal system L_f is *Prover9*, which is based on first-order logic syntax and a special signature of relevant predicates (e.g., *Square*, *FourSided*, *Shape*). The reasoning apparatus for this language is instantiated by *Prover9*; it is a sound calculus capable of verifying logical entailments within this formal system.

Consider the following example from the FOLIO benchmark:

Context: *All squares have four sides. All four-sided things are shapes.*

Question: *Based on the above information, is the following statement true, false, or uncertain? All squares are shapes.*

Options: A) True B) False C) Uncertain

This informal input is translated into the following expression, consisting of three different sets of first-order logic formulas:

Predicates:

Square(x) — x is a square

FourSided(x) — x has four sides

Shape(x) — x is a shape

Premises:

$\forall x(\text{Square}(x) \rightarrow \text{FourSided}(x))$

$\forall x(\text{FourSided}(x) \rightarrow \text{Shape}(x))$

Conclusion:

$\forall x(\text{Square}(x) \rightarrow \text{Shape}(x))$

Prover9 then verifies that the conclusion logically follows from the premises via transitive implication. Hence, the system formally determines that the correct answer is **True (A)**. In the paper, the semantic equivalence criterion E relies on all relevant properties of the objects described in the question being captured by the logical formulas, but without introducing spurious properties. The validation criterion V used to measure semantic equivalence consisted in comparing the answer derived from reasoning on the formalized problem to the ground truth correct answer. This approach enables automated evaluation. However, reaching the correct answer does not necessarily guarantee semantic equivalence.

Logic Programming The next example involves formalizing strategic interactions that can be modelled as bima-

trix games in order to reason about them formally in Prolog (Mensfelt, Stathis, and Trencsenyi 2025). In this case, L_f is the set of valid Prolog programs. The semantics of Prolog is well-defined and the reasoning apparatus relies on logical inference; in this case, it is realised by the SWI-Prolog solver.

In the following example, the relevant set of expressions involves description of game-like interactions, expressed in natural language (L_i). The specific example involves the classical Battle of the Sexes game:

A couple is deciding how to spend their evening together. One prefers to go to the opera and the other prefers to go to a football game, but both prefer to be together rather than apart. If they both choose the opera, the opera lover gets a payoff of 2 units of happiness, and the football lover gets 1 unit. If they both choose the football game, the football lover gets 2 units of happiness, and the opera lover gets 1 unit. If they choose different events, they both get 0 units of happiness.

The formalization of this expression results produces a set of domain-dependent predicates that, together with domain-independent predicates, allow for reasoning about the interaction. The snippet below, which is part of the formalization output, shows the payoff matrix for the game and the subset of predicates defining the initial state, legal moves and the effects of moves:

```
...
payoffBOS(opera, opera, 2, 1).
payoffBOS(football, football, 1, 2).
payoffBOS(opera, football, 0, 0).
payoffBOS(football, opera, 0, 0).

initial(s0).

initially(player(p1), s0).
initially(player(p2), s0).
initially(role(p1, opera_lover), s0).
initially(role(p2, football_lover), s0).
...
initial(s0).
...
legal(select(P, opera), S):-
    holds(player(P), S),
    holds(control(P), S).
...
effect(did(P, M), select(P, M), S).
...
abnormal(control(P), select(P, M), S).
...
```

The semantic equivalence criterion refers to the correct formalization of game rules and parameters (i.e., payoffs), such that player 1 selecting opera and player 2 selecting football should result in payoffs of (0,0) when we query the generated formalization. The verification V of the semantic equivalence criterion is automated by querying all possible combinations of actions and comparing the resulting payoffs to ground-truth payoffs for each pair of actions.

This method of evaluation is both automated and guarantees practical semantic equivalence. However, it applies

only to a narrow class of problems—bimatrix games—and requires manually implementing a validation procedure for each class of games.

Planning For planning we chose as an example the one used in (Oswald et al. 2024). In this case, the formal language L_f consists of PDDL actions. PDDL is a formal language for representing classical planning problems, where states are described using first-order logic atoms, actions define state transitions, and the goal is to find a sequence of actions that transforms the initial state into one satisfying goal conditions. The syntactic framework of PDDL defines the structure of predicates, object types, and action definitions with preconditions, and effects. For actions, the syntax consists of ‘(:action <action-name> ...)’ where action parameters are defined, followed by the ‘precondition’ (the conditions that must hold before the action can be executed) and the ‘effect’ (the changes that result from executing the action). The formal reasoning apparatus for PDDL included the planners that interpret the generated PDDL domain and solve planning problems. Specifically, the planners used in the paper include the K* planner (Lee, Katz, and Sohrabi 2023) (part of the semantic verification checks the correctness of 100 alternative plans for each problem instance) for plan generation.

The example informal language input (L_i) is mostly natural language, extended with symbols that represent names that must appear in the corresponding formal expression. Note that this is aided by adding a list of allowed predicates to the informal action description prompt:

Allowed Predicates:

*(in-city ?loc - place ?city - city): a place loc is in a city.
 (at ?obj - physobj ?loc - place) : a physical object obj is at a place loc.
 (in ?pkg - package ?veh - vehicle) : a package pkg is in a vehicle veh.*

The action, “FLY-AIRPLANE” will fly an airplane from one airport to another. After the action, the airplane will be in the new location.

The output is a PDDL representation of the described action, where the given action’s preconditions and effects are formalized to model the planning domain:

```
(:action FLY-AIRPLANE
 :parameters (?airplane - airplane
 ?loc-from - airport ?loc-to - airport)
 :precondition (at ?airplane ?loc-from)
 :effect (and(not(at ?airplane ?loc-from))
 (at ?airplane ?loc-to)) )
```

As with logical entailment in first-order logic, planning also requires a notion of semantic equivalence. However, domain equivalence in planning involves action behaviour as well as purely logical consequence. To evaluate approximate semantic alignment, the authors use a *heuristic domain equivalence* test. For a reconstructed domain \mathbf{D}_o' generated by replacing an action a in the original domain \mathbf{D}_o , plan-based reasoning is used to determine equivalence:

- Let Π be a planning problem solvable under \mathbf{D}_o , and Π' its analog using \mathbf{D}_o' . Let P and P' be subsets of valid plans generated from Π and Π' , respectively.
- Each plan $\pi \in P$ is checked for validity under Π' , and each $\pi' \in P'$ is validated under Π , using the VAL tool (Howey, Long, and Fox 2004).
- If all cross-validations succeed, then $\text{Semantics}(L_f') \approx \text{Semantics}(L_f)$, and the output action is deemed to be semantically equivalent in terms of planning behaviour.

This procedure defines a relaxed but practical notion of equivalence suitable for evaluating model outputs in symbolic planning systems. As with logical reasoning, selecting a formally well-structured action that “looks right” is not sufficient: only through validation methods like this can the semantic faithfulness of the model’s output be determined.

Knowledge Representation One approach to formalizing domain knowledge involves translating natural language (NL) statements into OWL axioms. (Mateiu and Groza 2023) demonstrate how fine-tuned large language models can facilitate this translation by converting concise declarative sentences into OWL axioms—expressed in *Functional Syntax*, one of several equivalent styles to write OWL axioms. Their system uses a GPT-3 model trained on a dataset of 150 prompt–response pairs. The training data covers various ontology constructs, including class declarations, subclass hierarchies, object properties, cardinality restrictions, and disjointness.

The Web Ontology Language (OWL) is a formalism used to model knowledge about a domain in terms of *named individuals*, their individual properties (called *concept names* or *classes*), and the binary relationships between them (called *roles* or *object properties*). OWL includes logical constructors that allow the definition of complex concepts and role expressions from simpler ones. The formal language L_f in this setting is the set of OWL ontologies: a set of logical assertions, typically divided into three categories: *TBox assertions*, which express general class relationships such as subclassing or equivalence; *ABox assertions*, which describe facts about individual instances; and *RBox assertions*, which define properties of roles, such as transitivity or symmetry. They often include also optional declaration statements (which can be inferred). Key reasoning tasks include consistency checking (determining whether the ontology admits a model) and classification (computing the taxonomy of concept names).

In this setting, the semantic equivalence criterion requires that all information described in the informal description is correctly captured by the ontology. The verification is often established informally by ontology engineers, who manually verify that the ontologies capture all relevant knowledge. However, reasoning tools play a crucial role in this process by checking for consistency and ensuring that the ontology entails all expected consequences while avoiding unintended ones. Ontology development is often an iterative process, alternating addition of axioms and reasoning to assess their impact.

To illustrate the approach of (Mateiu and Groza 2023), we recapitulate the output of their system on the sentence *Anna*

and Lana are each other’s sisters:

```
Declaration(ObjectProperty(:has-sister))
Declaration(NamedIndividual(:Anna))
Declaration(NamedIndividual(:Lana))
ObjPropAssert(:has-sister :Anna :Lana)
ObjPropAssert(:has-sister :Lana :Anna)
```

The first statement identifies the role (binary predicate) *has_sister* and named individuals *Anna* and *Lana*. Then it generates two additional assertions to represent the symmetry implied in the sentence.

In this setting, the semantic equivalence criterion E requires that all relational information in the informal description be correctly captured in the OWL ontology. In practice, this cannot be formally measured, so the paper establishes a validation criterion V based on ontology engineers manually verifying that the ontology encodes all relevant knowledge. In general, however, such manual V can be supplemented with an automated component: assuming the informal knowledge is consistent—as is usually the case—automated reasoning tools can check consistency of the formalization. Furthermore, when the expected consequences and undesirable outcomes of the knowledge base are known and have admissible formalizations, ontology reasoners such as Hermit (Glimm et al. 2014) can be used to verify formally that the ontology both entails all intended consequences and avoiding unintended ones.

4 Challenges and Open Problems

While there is a growing body of work aimed at advancing autoformalization (Patel, Saha, and Flanigan 2023; Zhang, Quan, and Freitas 2024; Li et al. 2024; Tarrach et al. 2024; Chan et al. 2025; Jiang, Li, and Jamnik 2024; Lu et al. 2025; Poiroux et al. 2024b; Liu et al. 2025b), several fundamental challenges still offer exciting research opportunities that will shape the field’s future development. We next provide a selective overview of these challenges and opportunities.

Semantic Verification The key challenge in autoformalization remains verifying equivalence of the informal language and the formal representation. Current approaches rely mostly on human supervision, either through direct assessment or comparison with reference solutions; this is sometimes enhanced with automated checks such as consistency checking of the extracted formalism. Fully formal and automated verification of semantic equivalence, however, appears impossible even in principle, since natural language and its intended meaning are inherently informal.

An open question is whether it is possible to train intelligent systems to acquire the dual competence that humans possess: understanding both formal and informal domains, and being able to assess correspondence between them. Such verification capabilities are in principle distinct from the current practice of autoformalization, but they could be integrated directly into the autoformalization process rather than applied post-hoc. Although formally unverifiable, using meta-level reasoning to relate formal and informal representations could significantly reduce supervi-

sion and move us closer to automated human-level semantic verification.

Target Formal Languages Another challenge in autoformalization is selecting an appropriate target formal language for the required task. This choice depends critically on the intended use case: who will interact with the formalized text, and what computational tasks will be performed with it? For instance, if the goal is interactive theorem proving, the availability and sophistication of verification tools as well as the contents of formal proof libraries become critical. On the other hand, if human mathematicians will work with the output, considerations of readability and established community conventions become essential.

The selection process must also balance expressiveness against computational tractability. The target language must be sufficiently expressive to capture all the relevant semantic content of the natural language text. However, there exists a fundamental tension between expressive power and computational complexity of verification, potentially making automated reasoning and verification very difficult or even undecidable. Given these constraints, an open problem is exploring the development of new formal languages designed specifically for autoformalization, which might offer better balance points in the expressiveness-tractability spectrum.

Scalability and Integration with Mathematical Practice Current autoformalization systems work well on curated benchmarks but often fail to scale to real-world practice. For example, in the context of mathematical proof, the literature is dominated by incomplete proofs, informal reasoning steps, and proof sketches, and so their formalization requires contextual understanding and gap-filling, capabilities that current systems struggle with, particularly in specialized mathematical fields.

This scalability gap reflects a deeper uncertainty about how autoformalization tools should integrate into research workflows. Without clarity on whether autoformalization should assist proof development, literature review, verification, hypothesis testing, or enhancing the reliability of AI systems, we risk building systems that are technical achievements but remain peripheral to the work of intended users.

Interactive Formalization Many of the current autoformalization approaches treat the task as a one-shot translation problem, attempting to convert natural language text directly into formal representations. As we have already discussed, this paradigm struggles with the inherent ambiguity and context-dependence of natural or mathematical language. A more promising direction—and an approach to the problem of semantic verification—involves systems that can engage in dialogue with users, asking clarifying questions about ambiguous terms, implicit assumptions, or intended interpretations.

Such interactive systems would require significant advances in both natural language understanding and theorem prover interfaces. The system should recognise when clarification is needed and formulate meaningful questions to the user, which then would be incorporated to iteratively refine the formalization. Early feedback would also prevent the

system from pursuing incorrect interpretations too far before correction.

Cross-Domain Transfer and Generalization As we have shown in this paper, current autoformalization approaches are often domain-specific, with systems designed separately for areas such as mathematical reasoning, planning, logic, etc. A major challenge is developing methods that can transfer reasoning patterns and formalization strategies across these different areas. Our present analysis of the notion of autoformalization shows that the underlying formal structures and reasoning patterns in these areas often share commonalities that could be exploited for more general autoformalization systems.

This, however, is still a big open challenge that requires a more systematic understanding of the structural similarities in the way autoformalization is used across fields. Success in this area would enable more robust and widely applicable autoformalization tools that could adapt to new domains with minimal additional training. We envision our current work, which establishes a common framework for the definition of autoformalization, as a first step in that direction.

5 Conclusions

In practice, we judge successful formalization by its fruits: Does it help us solve problems? Does it support verification? Does it reveal new connections? Does it eliminate disputes? Does it enable computation? Formalization is less about focusing on mere translation and more about creating tools that extend our and AI systems’ reasoning capabilities in useful ways.

One of the most promising applications is enhancing the capabilities of LLMs, developing robust general-purpose reasoners, and, as one of the consequences, enhancing the safety of agentic AI. However, achieving this through autoformalization presents many challenges. A common framework could accelerate progress by facilitating knowledge exchange between different fields and enabling transfer learning.

As we have demonstrated, translating problems into different formalisms for logical inference and automated reasoning shares similarities and common challenges. Based on the reviewed papers, we propose a preliminary definition that encompasses various types of formalisms. This definition is not intended to be definitive; rather, our aim is to initiate a cross-disciplinary dialogue on how best to conceptualize autoformalization, articulate its desired properties, and promote effective interdisciplinary collaboration.

Acknowledgments

This work was supported by a Leverhulme Trust International Professorship Grant (LIP-2022-001).

References

Abolhasani, M. S., and Pan, R. 2024. Leveraging LLM for automated ontology extraction and knowledge graph generation. *CoRR* abs/2412.00608.

- Aghzal, M.; Plaku, E.; Stein, G. J.; and Yao, Z. 2025. A Survey on Large Language Models for Automated Planning.
- Agrawal, A.; Gadgil, S.; Goyal, N.; Narayanan, A.; and Tadipatri, A. 2022. Towards a Mathematics Formalisation Assistant using Large Language Models. *arXiv:2211.07524*.
- Azerbaiyev, Z.; Piotrowski, B.; Schoelkopf, H.; Ayers, E. W.; Radev, D.; and Avigad, J. 2023. ProofNet: Autoformalizing and Formally Proving Undergraduate-Level Mathematics.
- Blanchette, J.; Desharnais, M.; Paulson, L. C.; and Bartl, L. 2025. *Hammering Away A User's Guide to Sledgehammer for Isabelle/HOL*. <https://isabelle.in.tum.de/dist/Isabelle2025/doc/sledgehammer.pdf>: tutorial.
- Borazjanizadeh, N., and Piantadosi, S. T. 2024. Reliable reasoning beyond natural language. *arXiv preprint arXiv:2407.11373*.
- Borroto Santana, M.; Kareem, I.; and Ricca, F. 2024. Towards automatic composition of asp programs from natural language specifications. *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*.
- Brickley, D., and Guha, R. V. 2014. Rdf schema 1.1. <https://www.w3.org/TR/rdf-schema/>. W3C Recommendation.
- Brunello, A.; Ferrarese, R.; Geatti, L.; Marzano, E.; Montanari, A.; and Saccomanno, N. 2025. Evaluating llms capabilities at natural language to logic translation: A preliminary investigation. In *Proceedings of the Logic and Engineering of Natural Language Semantics (LENLS) Workshop 2025*.
- Caufield, J. H.; Hegde, H.; Emonet, V.; Harris, N. L.; Joachimiak, M. P.; Matentzoglou, N.; Kim, H.; Moxon, S. A. T.; Reese, J. T.; Haendel, M. A.; Robinson, P. N.; and Mungall, C. J. 2024. Structured prompt interrogation and recursive extraction of semantics (SPIRES): a method for populating knowledge bases using zero-shot learning. *Bioinform.* 40(3).
- Chan, W.; Souliman, M.; Nordhagen, J.; Miranda, B.; Obbad, E.; and Koyejo, K. F. S. 2025. Lean-ing on Quality: How High-Quality Data Beats Diverse Multilingual Data in AutoFormalization.
- Chaturvedi, A., and Asher, N. 2024. Learning semantic structure through first-order-logic translation. *arXiv preprint arXiv:2410.03203*.
- Chen, Y.; Gandhi, R.; Zhang, Y.; and Fan, C. 2023. Nl2tl: Transforming natural languages to temporal logics using large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 15880–15903.
- Clarke, E. M.; Emerson, E. A.; and Sistla, A. P. 1986. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems* 8(2):244–263.
- Coppolillo, E.; Calimeri, F.; Manco, G.; Perri, S.; and Ricca, F. 2024. Lasp: Fine-tuning large language models for answer set programming. *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning*.
- Cunningham, G.; Bunesco, R. C.; and Juedes, D. 2023. Towards Autoformalization of Mathematics and Code Correctness: Experiments with Elementary Proofs.
- de la Rosa, T.; Gopalakrishnan, S.; Pozanco, A.; Zeng, Z.; and Borrajo, D. 2024. TRIP-PAL: Travel Planning with Guarantees by Combining Large Language Models and Automated Planners.
- Doumanas, D.; Soularidis, A.; Kotis, K.; and Vouros, G. A. 2024. Integrating llms in the engineering of a SAR ontology. In Maglogiannis, I.; Iliadis, L. S.; MacIntyre, J.; Avlonitis, M.; and Papaleonidas, A., eds., *Artificial Intelligence Applications and Innovations - 20th IFIP WG 12.5 International Conference, AIAI 2024, Corfu, Greece, June 27-30, 2024, Proceedings, Part IV*, volume 714 of *IFIP Advances in Information and Communication Technology*, 360–374. Springer.
- Eells, A.; Dave, B.; Hitzler, P.; and Shimizu, C. 2024. Commonsense ontology micropatterns. In Besold, T. R.; d'Avila Garcez, A.; Jiménez-Ruiz, E.; Confalonieri, R.; Madhyastha, P.; and Wagner, B., eds., *Neural-Symbolic Learning and Reasoning - 18th International Conference, NeSy 2024, Barcelona, Spain, September 9-12, 2024, Proceedings, Part II*, volume 14980 of *Lecture Notes in Computer Science*, 51–59. Springer.
- Fox, M., and Long, D. 2003. Pddl2. 1: An extension to pddl for expressing temporal planning domains. *Journal of artificial intelligence research* 20:61–124.
- Gadgil, S.; Tadipatri, A. R.; Agrawal, A.; Narayanan, A.; and Goyal, N. 2022. Towards automating formalisation of theorem statements using large language models. In *36th Conference on Neural Information Processing Systems (NeurIPS 2022) Workshop on MATH-AI*.
- Ghalab, M.; Howe, A.; Knoblock, C.; McDermott, I. D.; Ram, A.; Veloso, M.; Weld, D.; Sri, D. W.; Barrett, A.; Christianson, D.; et al. 1998. Pddl—the planning domain definition language. *IPC-98 Technical Report*.
- Glimm, B.; Horrocks, I.; Motik, B.; Stoilos, G.; and Wang, Z. 2014. Hermit: An owl 2 reasoner. *Journal of Autom. Reason.* 53(3):245–269.
- Guan, L.; Valmeekam, K.; Sreedharan, S.; and Kambhampati, S. 2023. Leveraging pre-trained large language models to construct and utilize world models for model-based task planning. *Advances in Neural Information Processing Systems* 36:79081–79094.
- Hahn, C.; Schmitt, F.; Tillman, J. J.; Metzger, N.; Siber, J.; and Finkbeiner, B. 2022. Formal Specifications from Natural Language.
- Han, S.; Schoelkopf, H.; Zhao, Y.; Qi, Z.; Riddell, M.; Zhou, W.; Coady, J.; Peng, D.; Qiao, Y.; Benson, L.; Sun, L.; Wardle-Solano, A.; Szabo, H.; Zubova, E.; Burtell, M.; Fan, J.; Liu, Y.; Wong, B.; Sailor, M.; Ni, A.; Nan, L.; Kasai, J.; Yu, T.; Zhang, R.; Fabbri, A. R.; Kryscinski, W.; Yavuz, S.; Liu, Y.; Lin, X. V.; Joty, S.; Zhou, Y.; Xiong, C.; Ying, R.; Cohan, A.; and Radev, D. 2022. FOLIO: Natural Language Reasoning with First-Order Logic.

- Helmert, M. 2006. The fast downward planning system. *Journal of Artificial Intelligence Research* 26:191–246.
- Howey, R.; Long, D.; and Fox, M. 2004. Val: Automatic plan validation, continuous effects and mixed-initiative planning using pddl. In *16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*.
- Hu, M.; Chen, T.; Zou, Y.; Lei, Y.; Chen, Q.; Li, M.; Mu, Y.; Zhang, H.; Shao, W.; and Luo, P. 2025a. Text2World: Benchmarking Large Language Models for Symbolic World Model Generation.
- Hu, X.; Zhou, Q.; Grechuk, B.; and Tyukin, I. Y. 2025b. Stepproof: Step-by-step verification of natural language mathematical proofs. *arXiv preprint arXiv:2506.10558*.
- Huang, C., and Zhang, L. 2024. On the Limit of Language Models as Planning Formalizers.
- Ishay, A.; Yang, Z.; and Lee, J. 2023. Leveraging large language models to generate answer set programs. *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning*.
- Jiang, A. Q.; Welleck, S.; Zhou, J. P.; Li, W.; Liu, J.; Jamnik, M.; Lacroix, T.; Wu, Y.; and Lample, G. 2022. Draft, sketch, and prove: Guiding formal theorem provers with informal proofs. *arXiv preprint arXiv:2210.12283*.
- Jiang, A. Q.; Li, W.; and Jamnik, M. 2024. Multi-language diversity benefits autoformalization. *Advances in Neural Information Processing Systems* 37:83600–83626.
- Jiang, Q. 2025. Language models for verifiable mathematical automation: Interaction, integration, and autoformalization.
- Kaliszyk, C.; Urban, J.; Vyskočil, J.; and Geuvers, H. 2014. Developing corpus-based translation methods between informal and formal mathematics: Project description. In *International Conference on Intelligent Computer Mathematics*, 435–439. Springer.
- Kaliszyk, C.; Urban, J.; and Vyskočil, J. 2015. Learning to parse on aligned corpora (rough diamond). In *Interactive Theorem Proving: 6th International Conference, ITP 2015, Nanjing, China, August 24-27, 2015, Proceedings* 6, 227–233. Springer.
- Kaliszyk, C.; Urban, J.; and Vyskočil, J. 2017. Automating formalization by statistical and semantic parsing of mathematics. In *International Conference on Interactive Theorem Proving*, 12–27. Springer.
- Kamath, A., and Das, R. 2018. A survey on semantic parsing. *arXiv preprint arXiv:1812.00978*.
- Karatarakis, M. 2024. Leveraging large language models for autoformalizing theorems: A case study. In *9th Conference on Artificial Intelligence and Theorem Proving (AITP)*.
- Kowalski, R. 1974. Predicate logic as a programming language. In *IFIP Congress*, 569–574.
- Lalwani, A.; Kim, T.; Chopra, L.; Hahn, C.; Jin, Z.; and Sachan, M. 2024. Autoformalizing Natural Language to First-Order Logic: A Case Study in Logical Fallacy Detection.
- Lee, J.; Lee, D.; Choi, C.; Im, Y.; Wi, J.; Heo, K.; Oh, S.; Lee, S.; and Shin, I. 2025. Safeguarding Mobile GUI Agent via Logic-based Action Verification.
- Lee, J.; Katz, M.; and Sohrabi, S. 2023. On k^* search for top-k planning. In *Proceedings of the International Symposium on Combinatorial Search*, volume 16, 38–46.
- Li, J., and Tian, M. T. 2025. Automatic generation of safety-compliant linear temporal logic via large language model: A self-supervised framework. *arXiv preprint arXiv:2503.15840*.
- Li, Z.; Wu, Y.; Li, Z.; Wei, X.; Zhang, X.; Yang, F.; and Ma, X. 2024. Autoformalize Mathematical Statements by Symbolic Equivalence and Semantic Consistency.
- Li, N.; Liu, P.; Liu, Z.; Dai, T.; Jiang, Y.; and Xia, S.-T. 2025. Logic-of-thought: Empowering large language models with logic programs for solving puzzles in natural language. *arXiv preprint arXiv:2505.16114*.
- Lin, K.; Agia, C.; Migimatsu, T.; Pavone, M.; and Bohg, J. 2023. Text2Motion: from natural language instructions to feasible plans. *Autonomous Robots* 47(8):1345–1365.
- Lippolis, A. S.; Saeedizade, M. J.; Keski-Särkkä, R.; Zuppiroli, S.; Ceriani, M.; Gangemi, A.; Blomqvist, E.; and Nuzzolese, A. G. 2025. Ontology generation using large language models. In Curry, E.; Acosta, M.; Poveda-Villalón, M.; van Erp, M.; Ojo, A. K.; Hose, K.; Shimizu, C.; and Lisena, P., eds., *The Semantic Web - 22nd European Semantic Web Conference, ESWC 2025, Portoroz, Slovenia, June 1-5, 2025, Proceedings, Part I*, volume 15718 of *Lecture Notes in Computer Science*, 321–341. Springer.
- Liu, B.; Jiang, Y.; Zhang, X.; Liu, Q.; Zhang, S.; Biswas, J.; and Stone, P. 2023. LLM+P: Empowering Large Language Models with Optimal Planning Proficiency.
- Liu, J. X.; Shah, A.; Konidaris, G.; Tellex, S.; and Paulius, D. 2024a. Lang2LTL-2: Grounding Spatiotemporal Navigation Commands Using Large Language and Vision-Language Models. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2325–2332. ISSN: 2153-0866.
- Liu, Q.; Zheng, X.; Lu, X.; Cao, Q.; and Yan, J. 2024b. Rethinking and Improving Autoformalization: Towards a Faithful Metric and a Dependency Retrieval-based Approach. In *Proceedings of The Thirteenth International Conference on Learning Representations*.
- Liu, C.; Yuan, Y.; Yin, Y.; Xu, Y.; Xu, X.; Chen, Z.; Wang, Y.; Shang, L.; Liu, Q.; and Zhang, M. 2025a. Safe: Enhancing mathematical reasoning in large language models via retrospective step-aware formal verification. *arXiv preprint arXiv:2506.04592*.
- Liu, Q.; Zheng, X.; Lu, X.; Cao, Q.; and Yan, J. 2025b. Rethinking and improving autoformalization: Towards a faithful metric and a dependency retrieval-based approach. In *The Thirteenth International Conference on Learning Representations (ICLR)*. Spotlight paper.
- Liu, X.; Bao, K.; Zhang, J.; Liu, Y.; Chen, Y.; Liu, Y.; Jiao, Y.; and Luo, T. 2025c. ATLAS: Autoformalizing Theorems through Lifting, Augmentation, and Synthesis of Data.

- Liu, J. 2025. Few-shot natural language to first-order logic translation via in-context learning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 10939–10960.
- Lu, J.; Wan, Y.; Liu, Z.; Huang, Y.; Xiong, J.; Liu, C.; Shen, J.; Jin, H.; Zhang, J.; Wang, H.; Yang, Z.; Tang, J.; and Guo, Z. 2024. Process-Driven Autoformalization in Lean 4.
- Lu, J.; Wan, Y.; Huang, Y.; Xiong, J.; Liu, Z.; and Guo, Z. 2025. Formalalign: Automated alignment evaluation for autoformalization. In *The Thirteenth International Conference on Learning Representations (ICLR)*. Preprint accepted to ICLR 2025.
- Mahdavi, S.; Aoki, R.; Tang, K.; and Cao, Y. 2024. Leveraging Environment Interaction for Automated PDDL Translation and Planning with Large Language Models.
- Mateiu, P., and Groza, A. 2023. Ontology engineering with large language models. In *25th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2023, Nancy, France, September 11-14, 2023*, 226–229. IEEE.
- Mavrogiannis, A.; Mavrogiannis, C.; and Aloimonos, Y. 2024. Cook2LTL: Translating Cooking Recipes to LTL Formulae using Large Language Models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 17679–17686.
- McCarthy, J. 1960. Programs with common sense. Technical report, USA.
- Mensfelt, A.; Stathis, K.; and Trencsenyi, V. 2024. Autoformalization of game descriptions using large language models.
- Mensfelt, A.; Stathis, K.; and Trencsenyi, V. 2025. Generative agents for multi-agent autoformalization of interaction scenarios. Accepted at European Conference on Artificial Intelligence (ECAI) 2025.
- Murphy, L.; Yang, K.; Sun, J.; Li, Z.; Anandkumar, A.; and Si, X. 2024. Autoformalizing Euclidean Geometry.
- Newell, A., and Simon, H. A. 1976. Computer science as empirical inquiry: Symbols and search. *Communications of the ACM* 19(3):113–126.
- Nguyen, H. T.; Fungwacharakorn, W.; Zin, M. M.; Goebel, R.; Toni, F.; Stathis, K.; and Satoh, K. 2025. LLMs for legal reasoning: A unified framework and future perspectives. *Computer Law & Security Review* 58:106165.
- Nipkow, T.; Wenzel, M.; and Paulson, L. C. 2002. *Isabelle/HOL: a proof assistant for higher-order logic*. Springer.
- Oates, T.; Alford, R.; Johnson, S.; and Hall, C. 2024. Using large language models to extract planning knowledge from unstructured text. In *Proceedings of the ICAPS Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.
- Olausson, T.; Gu, A.; Lipkin, B.; Zhang, C.; Solar-Lezama, A.; Tenenbaum, J.; and Levy, R. 2023. LINC: A Neurosymbolic Approach for Logical Reasoning by Combining Language Models with First-Order Logic Provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 5153–5176. Singapore: Association for Computational Linguistics.
- Oswald, J.; Srinivas, K.; Kokel, H.; Lee, J.; Katz, M.; and Sohrabi, S. 2024. Large Language Models as Planning Domain Generators. *Proceedings of the International Conference on Automated Planning and Scheduling* 34:423–431.
- OWL Working Group. 2012. Owl 2 web ontology language document overview (second edition). W3c recommendation, World Wide Web Consortium (W3C). Accessed: 2025-07-28.
- Pan, L.; Albalak, A.; Wang, X.; and Wang, W. Y. 2023. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.
- Pan, J.; Chou, G.; and Berenson, D. 2023. Data-efficient learning of natural language to linear temporal logic translators for robot task specification. *Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA)*.
- Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 311–318.
- Patel, N.; Saha, R.; and Flanagan, J. 2023. A New Approach Towards Autoformalization.
- Pnueli, A. 1977. The temporal logic of programs. *18th Annual Symposium on Foundations of Computer Science* 46–57.
- Poiroux, A.; Weiss, G.; Kunčák, V.; and Bosselut, A. 2024a. Improving autoformalization using type checking. *arXiv preprint arXiv:2406.07222*.
- Poiroux, A.; Weiss, G.; Kunčák, V.; and Bosselut, A. 2024b. Improving Autoformalization using Type Checking.
- Quan, X.; Valentino, M.; Dennis, L. A.; and Freitas, A. 2024. Verification and Refinement of Natural Language Explanations through LLM-Symbolic Theorem Proving.
- Ryu, H.; Kim, G.; Lee, H. S.; and Yang, E. 2024. Divide and Translate: Compositional First-Order Logic Translation and Verification for Complex Logical Reasoning.
- Saeedizade, M. J., and Blomqvist, E. 2024. Navigating ontology development with large language models. In Meroño-Peñuela, A.; Dimou, A.; Troncy, R.; Hartig, O.; Acosta, M.; Alam, M.; Paulheim, H.; and Lisena, P., eds., *The Semantic Web - 21st International Conference, ESWC 2024, Heraklion, Crete, Greece, May 26-30, 2024, Proceedings, Part I*, volume 14664 of *Lecture Notes in Computer Science*, 143–161. Springer.
- Shojaee, P.; Mirzadeh, I.; Alizadeh, K.; Horton, M.; Bengio, S.; and Farajtabar, M. 2025. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. *arXiv preprint arXiv:2506.06941*.

- Sikes, K.; Fine-Morris, M.; Sreedharan, S.; Smith, L. N.; and Roberts, M. 2025. Creating PDDL Models from Javascript using LLMs: Preliminary Results. In *AAAI 2025 Workshop LM4Plan*.
- Silver, T.; Dan, S.; Srinivas, K.; Tenenbaum, J. B.; Kaelbling, L.; and Katz, M. 2024. Generalized Planning in PDDL Domains with Pretrained Large Language Models. *Proceedings of the AAAI Conference on Artificial Intelligence* 38(18):20256–20264.
- Smirnov, P.; Joubin, F.; Ceravola, A.; and Gienger, M. 2024. Generating consistent PDDL domains with Large Language Models.
- Soroco, M.; Song, J.; Xia, M.; Emond, K.; Sun, W.; and Chen, W. 2025. Pde-controller: LLMs for autoformalization and reasoning of pdes. *arXiv preprint arXiv:2502.00963*.
- Szegedy, C. 2020. A promising path towards autoformalization and general artificial intelligence. In *Intelligent Computer Mathematics: 13th International Conference, CICM 2020, Bertinoro, Italy, July 26–31, 2020, Proceedings 13*, 3–20. Springer.
- Tang, Y.; da Costa, A. A. B.; Zhang, X.; Irvine, P.; Khastgir, S.; and Jennings, P. A. 2023. Domain knowledge distillation from large language model: An empirical study in the autonomous driving domain. In *26th IEEE International Conference on Intelligent Transportation Systems, ITSC 2023, Bilbao, Spain, September 24–28, 2023*, 3893–3900. IEEE.
- Tarrach, G.; Jiang, A. Q.; Raggi, D.; Li, W.; and Jamnik, M. 2024. More details, please: Improving autoformalization with more detailed proofs. In *AI for Math Workshop@ ICML 2024*.
- Thatikonda, R. K.; Han, J.; Buntine, W.; and Shareghi, E. 2024. Strategies for Improving NL-to-FOL Translation with LLMs: Data Generation, Incremental Fine-Tuning, and Verification.
- Wang, Q.; Brown, C.; Kaliszyk, C.; and Urban, J. 2020. Exploration of neural machine translation in autoformalization of mathematics in mizar. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, 85–98.
- Wang, H.; Unsal, M.; Lin, X.; Baksys, M.; Liu, J.; Santos, M. D.; Sung, F.; Vinyes, M.; Ying, Z.; Zhu, Z.; et al. 2025a. Kimina-prover preview: Towards large formal reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.11354*.
- Wang, J.; Sundarsingh, D. S.; Deshmukh, J. V.; and Kantaros, Y. 2025b. ConformalNL2LTL: Translating Natural Language Instructions into Temporal Logic Formulas with Conformal Correctness Guarantees. *arXiv:2504.21022*.
- Wang, R.; Li, Y.; Fung, Y. R. M.; and Zhang, T. 2025c. Let’s reason formally: Natural-formal hybrid reasoning enhances llm’s math capability. *arXiv preprint arXiv:2505.23703*.
- Wang, Q.; Kaliszyk, C.; and Urban, J. 2018. First experiments with neural translation of informal to formal mathematics. In *Intelligent Computer Mathematics: 11th International Conference, CICM 2018, Hagenberg, Austria, August 13–17, 2018, Proceedings 11*, 255–270. Springer.
- Warren, D. H., and Pereira, F. C. 1982. An efficient easily adaptable system for interpreting natural language queries. *American Journal of Computational Linguistics* 8(3-4):110–122.
- Weizenbaum, J. 1966. Eliza — a computer program for the study of natural language communication between man and machine. *Communications of the ACM* 9(1):36–45.
- Weng, K.; Du, L.; Li, S.; Lu, W.; Sun, H.; Liu, H.; and Zhang, T. 2025. Autoformalization in the era of large language models: A survey. *arXiv preprint arXiv:2505.23486*.
- William, M.; Nikolaus, H.; Koenig, N.; Leyi, C.; Rothkopf, R.; Qiao, F.; and Santolucito, M. 2024. Guiding llm temporal logic generation with explicit separation of data and control. *arXiv preprint arXiv:2406.07400*.
- Winograd, T. 1972. Understanding natural language. *Cognitive Psychology* 3(1):1–191.
- Wu, Y.; Jiang, A. Q.; Li, W.; Rabe, M.; Staats, C.; Jamnik, M.; and Szegedy, C. 2022. Autoformalization with Large Language Models. *Advances in Neural Information Processing Systems* 35:32353–32368.
- Xie, Y.; Yu, C.; Zhu, T.; Bai, J.; Gong, Z.; and Soh, H. 2023. Translating Natural Language to Planning Goals with Large-Language Models.
- Xu, Y.; Feng, J.; and Miao, W. 2024. Learning from Failures: Translation of Natural Language Requirements into Linear Temporal Logic with Large Language Models. In *2024 IEEE 24th International Conference on Software Quality, Reliability and Security (QRS)*, 204–215. ISSN: 2693-9177.
- Yang, K.; Poesia, G.; He, J.; Li, W.; Lauter, K.; Chaudhuri, S.; and Song, D. 2024. Formal mathematical reasoning: A new frontier in ai. *arXiv preprint arXiv:2412.16075*.
- Yang, Z.; Ishay, A.; and Lee, J. 2023. Coupling Large Language Models with Logic Programming for Robust and General Reasoning from Text.
- Ying, H.; Wu, Z.; Geng, Y.; Wang, J.; Lin, D.; and Chen, K. 2024. Lean Workbook: A large-scale Lean problem set formalized from natural language math problems.
- Yu, Z.; Peng, R.; Ding, K.; Li, Y.; Peng, Z.; Liu, M.; Zhang, Y.; Yuan, Z.; Xin, H.; Huang, W.; Wen, Y.; Zhang, G.; and Liu, W. 2025. Formalmath: Benchmarking formal mathematical reasoning of large language models.
- Zhang, L.; Quan, X.; and Freitas, A. 2024. Consistent autoformalization for constructing mathematical libraries. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Zhang, L.; Valentino, M.; and Freitas, A. 2025. Formalizing Complex Mathematical Statements with LLMs: A Study on Mathematical Definitions.
- Zhou, J. P.; Staats, C.; Li, W.; Szegedy, C.; Weinberger, K. Q.; and Wu, Y. 2024. Don’t Trust: Verify – Grounding LLM Quantitative Reasoning with Autoformalization.

Multi-Agent Legal Verifier Systems for Data Transfer Planning

Ha Thanh Nguyen^{1,2}, Wachara Fungwacharakorn¹, Ken Satoh¹

¹ Center of Juris-Informatics, Joint Support-Center for Data Science Research, ROIS, Tokyo, Japan

² Research and Development Center for Large Language Models, NII, ROIS, Tokyo, Japan

Abstract

Legal compliance in AI-driven data transfer planning is becoming increasingly critical under stringent privacy regulations such as the Japanese Act on the Protection of Personal Information (APPI). We propose a multi-agent legal verifier that decomposes compliance checking into specialized agents for statutory interpretation, business context evaluation, and risk assessment, coordinated through a structured synthesis protocol. Evaluated on a stratified dataset of 200 APPI Article 16 cases with clearly defined ground truth labels and multiple performance metrics, the system achieves 72% accuracy, which is 21 percentage points higher than a single-agent baseline, including 90% accuracy on clear compliance cases (vs. 16% for the baseline) while maintaining perfect detection of clear violations. While challenges remain in ambiguous scenarios, these results show that domain specialization and coordinated reasoning can meaningfully improve legal AI performance, providing a scalable and regulation-aware framework for trustworthy and interpretable automated compliance verification.

1 Introduction

The proliferation of artificial intelligence systems in data-intensive applications has created an urgent need for automated legal compliance verification. As organizations increasingly rely on AI agents for data transfer planning and execution, ensuring adherence to privacy regulations has become a critical challenge. This requires embedding legal reasoning (Nguyen et al. 2025) as a core capability within AI decision-making processes. The Japanese Act on the Protection of Personal Information (APPI), particularly Article 16 concerning Purpose Limitation, exemplifies the complex regulatory landscape that AI systems must navigate.

Traditional approaches to legal compliance checking typically employ single-agent architectures that attempt to capture the full complexity of legal reasoning within a monolithic system. However, legal compliance involves multiple dimensions of analysis: legal interpretation, contextual understanding, risk assessment, and precedent consideration. This multifaceted nature suggests that specialized multi-agent approaches may offer superior performance.

This paper introduces a novel multi-agent legal verifier system designed specifically for data transfer planning scenarios. Our approach transforms high-level plans into lists of specific actions, each of which is systematically evaluated

by specialized legal verifier agents. The core innovation lies in the decomposition of legal compliance checking into specialized roles: legal analysis, context evaluation, and risk assessment, coordinated by a decision-making agent.

We present a comprehensive experimental evaluation comparing single-agent and multi-agent approaches across 200 carefully constructed test cases representing diverse compliance scenarios under APPI Article 16. Our results demonstrate significant performance improvements for the multi-agent system, particularly in complex compliance scenarios requiring nuanced legal reasoning.

The contributions of this work are threefold: (1) a novel multi-agent architecture for legal compliance verification in data transfer planning, (2) a comprehensive experimental framework for evaluating legal AI systems, and (3) empirical evidence demonstrating the superiority of multi-agent approaches for complex legal reasoning tasks.

The rest of this paper is organized as follows. Section 2 reviews prior work on applying knowledge representation, multi-agent systems, and subsymbolic AI to legal compliance. Section 3 provides the formalization of data transfer planning, APPI Article 16 compliance, and the legal verification problem. Section 4 details the proposed system. Section 5 outlines the experimental design. Section 6 reports the results and the detailed analysis. Section 7 discusses our findings. Finally, Section 8 concludes the paper.

2 Related Work

Legal compliance has been explored across multiple areas of AI, including knowledge representation, multi-agent systems, and subsymbolic approaches. This section reviews previous work in each of these areas.

2.1 Knowledge Representation and Legal Compliance

Researchers have long sought to formalize legal compliance using various forms of logic. Because legal compliance inherently involves deontic concepts – such as obligations, prohibitions, and permissions – it is a natural application of deontic logic (Von Wright 1951). It also requires temporal logic (Alur, Courcoubetis, and Dill 1990) to capture the initiation and termination of actions, regulations, and their effects. Moreover, real-world legal systems contain exceptions and conflicts, making defeasible logic (Nute

1994) a key foundation for modeling legal compliance. To achieve more practical formalizations, researchers have also explored combinations of these logics, including defeasible deontic logic (Nute 1997) and temporal defeasible logic (Governatori and Terenziani 2007).

Building on these logical foundations, several first-order knowledge representations have been developed to formalize legal compliance (Robaldo et al. 2024). PROLEG (Satoh et al. 2010) is a Prolog-based framework that models exceptions explicitly, rather than using negation as failure, to better simulate legal reasoning. LegalRuleML (Palmirani et al. 2011) is an XML-based representation that supports conflict resolution through superiority relations. Institution Action Language (Padget et al. 2016) is a declarative language designed to represent institutional actions and deontic concepts.

Recent research has focused on complex scenarios such as business process compliance. Modeling business processes often requires semantic annotations (e.g., “control tags” or “flow tags”) to link tasks with compliance requirements. The process typically involves generating execution traces, identifying triggered obligations, and checking for fulfillment, violations, or compensations (Sadiq and Governatori 2014). Due to its declarative nature and suitability for automating compliance checking over complex scenarios and event traces, answer set programming (ASP) has been widely applied in this domain (Cliffe, De Vos, and Padget 2006; Arias et al. 2024).

In addition, researchers have investigated methods for explaining legal compliance and facilitating information sharing. Argumentation is a prominent approach to explaining compliance decisions (Burgemeestre, Hulstijn, and Tan 2011; Prakken and Sartor 2015). Several systematic verification methods for information sharing have also been explored (Amor and Dimyadi 2021), though primarily in the context of building codes rather than data protection regulations. Nonetheless, the principles of automated verification in these domains offer valuable insights for the design of legal compliance systems.

2.2 Multi-Agent Systems and Legal Compliance

Researchers have extensively examined legal compliance in multi-agent systems, particularly focusing on decentralizing compliance checking. Steenhuisen et al. (2006) demonstrated how multi-agent planning frameworks can manage complex coordination tasks among non-cooperative agents by formalizing task decomposition, modeling inter-agent dependencies, and analyzing coordination complexity – insights directly applicable to legal compliance. Alechina et al. (2016) introduced a decentralized norm-monitoring framework for open multi-agent environments, demonstrating how agents can effectively detect and address norm violations without relying on a centralized authority. Their approach leverages an incentive-based mechanism in which agents both perform their primary tasks and monitor each other’s behaviour, with monitoring costs covered through a scrip system. This design enables the achievement of perfect monitoring at equilibrium or the adjustment of monitoring frequency to maintain low violation probabilities, high-

lighting the importance of allocating specialized monitoring responsibilities across different compliance dimensions.

Other studies have explored legal compliance in different multi-agent contexts, such as multi-level governance and epistemic planning. King et al. (2017) examined multi-level governance compliance, designing institutional structures where higher-level institutions govern lower-level ones. Liu and Liu (2018) extended epistemic planning frameworks to handle common knowledge via a novel normal form in KD45 logic, and implemented the MEPC planner capable of generating solutions in domains requiring coordination through shared knowledge.

Recent advances in multi-agent systems have shown promising applications in combining legal and ethical compliance. Hayashi et al. (2023) proposed a multi-agent online planning architecture for real-time compliance, combining legal and ethical compliance mechanisms for trustworthy AI. Their work demonstrated the feasibility of integrating multiple specialized agents for regulatory compliance, though focused primarily on planning rather than verification.

2.3 Subsymbolic AI and Legal Compliance

With the rapid advancement of subsymbolic artificial intelligence, its implications for legal compliance have attracted growing attention. Mitrou (2018) offered a comprehensive analysis of the relationship between the GDPR and AI, especially subsymbolic AI, questioning whether current data protection regulations are truly “AI-proof.” This work emphasized the need for specialized systems capable of interpreting complex regulatory requirements in AI contexts. Javed and Li (2024) investigated semantic bias classification in legal judgments using subsymbolic AI, illustrating the potential of automated legal analysis while also identifying challenges in ensuring fairness and accuracy. Cerqueira (2024) examined the trustworthiness of LLM-based multi-agent systems for ethical AI, including GDPR compliance and fairness evaluation, highlighting both the promise and the limitations of current LLM approaches for legal reasoning.

To leverage the strengths and mitigate the weaknesses of both symbolic and subsymbolic AI, some researchers have proposed hybrid approaches. Nguyen et al. (Nguyen et al. 2023) extended the PROLEG logic-programming framework for Japanese legal reasoning by integrating deep learning techniques – such as fact extraction and end-to-end text interpretation – to enhance interpretability, feasibility, and alignment with practitioners’ needs in compliance-checking systems. Nguyen and Satoh (2024) also introduced the Krag (Knowledge Representation Augmented Generation) framework, which incorporates inference graphs via Soft PROLEG to guide LLMs in structured legal reasoning, thereby improving both precision and explainability in compliance-related outputs.

3 Problem Formulation

This section provides the formalization of data transfer planning, the Japanese Act on the Protection of Personal Infor-

mation (APPI) Article 16 compliance, and legal verifier problem.

3.1 Data Transfer Planning Context

We consider a data transfer planning scenario where an AI system must generate and execute plans for transferring personal information across different jurisdictions and systems. Each plan P consists of a sequence of actions $A = \{a_1, a_2, \dots, a_n\}$, where each action a_i involves the handling of personal information for a specific purpose.

Definition 1 (Data Transfer Action). *A data transfer action a_i is defined as a tuple (c, p, d, o) where:*

- c represents the company context and data handling capabilities
- p represents the stated purpose of data utilization
- d represents the proposed data handling operation
- o represents additional operational context

3.2 APPI Article 16 Compliance

APPI Article 16 imposes restrictions on handling personal information *beyond the Purpose of Utilization* specified under Article 15, unless explicit prior consent is obtained from the data subject.

1. **Primary Rule (Paragraph 1):** A business operator shall not handle personal information beyond the scope necessary for achieving the stated purpose, without prior consent of the individual.
2. **Business Succession Rule (Paragraph 2):** If personal information is acquired through business succession (e.g., merger, acquisition), the same restriction applies with respect to the original purpose prior to the succession.
3. **Exceptions (Paragraph 3):** The above restrictions do not apply when:
 - (a) Handling is based on laws or regulations
 - (b) Necessary to protect life, body, or property where obtaining consent is difficult
 - (c) Specially necessary for public health or the sound growth of children where obtaining consent is difficult
 - (d) Necessary to cooperate with governmental authorities in legal duties where obtaining consent may impede execution

Definition 2 (Compliance Status). *For a given action a_i , the compliance status $S(a_i) \in \{COMPLIANT, NON-COMPLIANT\}$ is determined by:*

$$S(a_i) = \begin{cases} COMPLIANT, & \text{if:} \\ & d \text{ is within purpose } p, \\ & \text{or prior consent is obtained,} \\ & \text{or an exception applies} \\ NON-COMPLIANT, & \text{otherwise.} \end{cases}$$

3.3 Legal Verifier Problem

The legal verifier problem can be formalized as a function $V : A \rightarrow S$ that maps each action to its compliance status. The challenge lies in accurately implementing this function

given the complexity and ambiguity inherent in legal interpretation.

Traditional single-agent approaches attempt to implement V directly through a monolithic reasoning system. Our multi-agent approach decomposes V into specialized sub-functions:

$$V(a_i) = C(L(a_i), X(a_i), R(a_i)) \quad (1)$$

where:

- $L(a_i)$ represents legal analysis of the action
- $X(a_i)$ represents contextual analysis of business necessity
- $R(a_i)$ represents risk assessment and edge case evaluation
- $C(\cdot)$ represents the coordination function that synthesizes these analyses

4 Multi-Agent Legal Verifier Architecture

The section proposes a design of **multi-agent legal verifier system**, including agents and coordination protocols.

4.1 Agent Specialization

Our multi-agent legal verifier system consists of four specialized agents working in coordination to evaluate legal compliance.

1. **Legal Analyst Agent:** Specializes in interpreting legal requirements and precedents
2. **Context Analyzer Agent:** Focuses on business context and purpose alignment
3. **Risk Assessor Agent:** Evaluates privacy risks and edge cases
4. **Coordinator Agent:** Synthesizes analyses and makes final decisions

Legal Analyst Agent The Legal Analyst Agent (L) focuses on pure legal interpretation, analyzing whether proposed actions align with statutory requirements. Its primary responsibilities include:

- Interpreting the scope of "necessary for the achievement of the Purpose of Utilization"
- Identifying when actions clearly fall within or outside legal boundaries
- Applying legal precedents and interpretative guidelines

The agent operates with high confidence on clear legal violations but may express uncertainty on borderline cases, providing valuable signals to the coordination process.

Context Analyzer Agent The Context Analyzer Agent (X) specializes in business context evaluation, determining whether proposed actions are reasonably necessary for stated business purposes. Key functions include:

- Evaluating business necessity and proportionality
- Assessing alignment between stated purposes and proposed actions

- Identifying legitimate business justifications for data handling

This agent provides crucial context that pure legal analysis might miss, particularly in complex business scenarios.

Risk Assessor Agent The Risk Assessor Agent (R) focuses on privacy risk evaluation and edge case identification. Its responsibilities include:

- Assessing privacy risks associated with proposed actions
- Evaluating consent mechanisms and their adequacy
- Identifying special circumstances that might affect compliance
- Analyzing potential consequences of data handling decisions

Coordinator Agent The Coordinator Agent (C) synthesizes the analyses from all specialist agents to make final compliance determinations. It employs sophisticated reasoning to:

- Weight different perspectives based on confidence levels
- Resolve conflicts between agent assessments
- Apply meta-reasoning about legal uncertainty
- Generate comprehensive justifications for decisions

4.2 Coordination Protocol

The multi-agent system follows a structured coordination protocol:

1. $analysis_L \leftarrow \text{LegalAnalyst.analyze}(a_i)$
2. $analysis_X \leftarrow \text{ContextAnalyzer.analyze}(a_i)$
3. $analysis_R \leftarrow \text{RiskAssessor.analyze}(a_i)$
4. $analyses \leftarrow \{analysis_L, analysis_X, analysis_R\}$
5. $(S(a_i), J(a_i)) \leftarrow \text{Coordinator.synthesize}(a_i, analyses)$

5 Experimental Design

This section describes the experiment design, including dataset construction, evaluation metrics, and baseline comparison.

5.1 Dataset Construction

We constructed a comprehensive dataset of 200 test cases representing diverse compliance scenarios under APPI Article 16. The dataset is stratified across four categories:

- **Clear Compliance** (50 cases): Actions clearly within stated purposes
- **Clear Violations** (50 cases): Actions clearly beyond stated purposes without consent
- **Consent-Based Compliance** (50 cases): Actions beyond original purpose with explicit consent
- **Edge Cases** (50 cases): Ambiguous scenarios requiring nuanced interpretation

Each test case includes:

- Company context and data handling capabilities

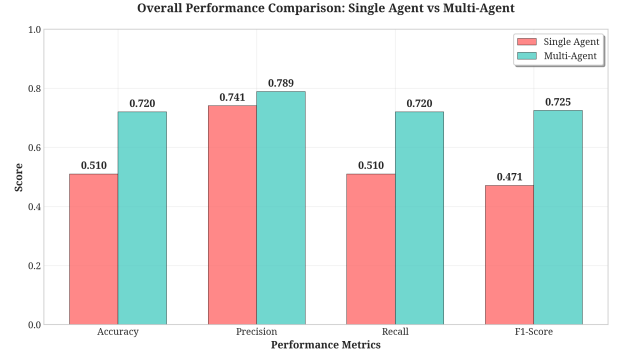


Figure 1: Overall Performance Comparison between Single Agent and Multi-Agent approaches across all evaluation metrics. The multi-agent system shows substantial improvements in accuracy, recall, and F1-score.

- Stated purpose of data utilization
- Proposed data handling action
- Additional operational context
- Ground truth compliance label with detailed reasoning

5.2 Evaluation Metrics

We evaluate system performance using standard classification metrics:

- **Accuracy:** Overall proportion of correct classifications
- **Precision:** Proportion of predicted compliant cases that are actually compliant
- **Recall:** Proportion of actually compliant cases correctly identified
- **F1-Score:** Harmonic mean of precision and recall

Additionally, we analyze:

- **Category-specific performance:** Accuracy within each compliance category
- **Confidence calibration:** Relationship between predicted confidence and actual accuracy
- **Processing time:** Computational efficiency comparison

5.3 Baseline Comparison

We compare our multi-agent system against a single-agent baseline that attempts to perform all aspects of legal analysis within a unified reasoning framework. Both systems use identical underlying language models (GPT-3.5-turbo) to ensure fair comparison.

6 Results and Analysis

This section reports the experiment results, including overall performance, category-specific analysis, confidence analysis, and processing time analysis.

6.1 Overall Performance

Table 1 presents the comprehensive comparison between single-agent and multi-agent approaches. The visualization in Figure 1 complements the tabular data, highlighting that the multi-agent system significantly outperforms the single-agent baseline across all key metrics.

The multi-agent system demonstrates substantial improvements across all metrics. Notably:

- **Accuracy** shows a 41.2% relative improvement, indicating better overall correctness of predictions.
- **F1-Score** gains 54.1%, reflecting a balanced enhancement in both precision and recall.
- **Precision** improves modestly (+6.5%), showing that the multi-agent approach retains the ability to avoid false positives.
- **Recall** matches the accuracy improvement, confirming that more relevant cases are correctly identified.

These results suggest that the multi-agent architecture enhances both the breadth (recall) and reliability (precision) of detection, with the greatest impact on balanced performance measures.

6.2 Category-Specific Analysis

Table 2 details the breakdown by compliance category, showing how performance gains vary depending on case characteristics.

Analysis reveals:

- **Clear Compliance:** Largest gain (+74.0 pp), suggesting that specialized agents are better at confirming legitimate, rule-abiding behavior.
- **Clear Violations:** Both approaches achieve perfect scores, implying that obvious breaches are trivial to detect regardless of architecture.
- **Consent-Based Compliance:** Moderate gain (+10.0 pp) points to improved handling of nuanced, consent-related decisions by leveraging specialized risk assessment.
- **Edge Cases:** No improvement observed, underscoring that truly ambiguous situations remain a challenge for both systems.

These patterns suggest that the multi-agent system delivers the most value in unambiguous compliance scenarios, with incremental benefits for moderately complex cases.

6.3 Confidence Analysis

Figure 3 compares how confidence scores map to actual prediction accuracy. Key findings:

- The multi-agent system’s confidence scores exhibit stronger correlation with true accuracy, indicating better calibration.
- High-confidence predictions from the multi-agent system are more trustworthy, reducing risk in high-stakes decisions.

- This improved calibration suggests that the system effectively synthesizes uncertainty estimates from specialized agents, resulting in a more informed and balanced final decision.

Such calibration improvements are crucial in operational contexts, where decision-makers rely on confidence scores to allocate review resources.

6.4 Processing Time Analysis

The multi-agent approach incurs a 6.67x processing time overhead (9.31s vs 1.39s average), as shown in Figure 4, reflecting the additional computational cost of specialized analysis and coordination. However, this overhead may be acceptable for high-stakes compliance decisions where accuracy is paramount.

7 Discussion

This section discusses several findings of this paper, including its implications, architectural insights, limitations, and future work.

7.1 Implications for Legal AI

Our findings reinforce the view that legal reasoning—particularly in the domain of compliance assessment—is not a monolithic problem, but a layered cognitive task that benefits from distributed expertise. The substantial 21.0 percentage point increase in accuracy achieved by the multi-agent system represents not merely an incremental improvement, but a meaningful shift in the feasibility of automated legal compliance checking.

This improvement carries direct implications for both the practice of law and the design of legal technology. In real-world deployments, such a system could serve as a front-line compliance filter, reducing the volume of manual reviews required by human legal teams. More importantly, the multi-agent architecture enables the AI to differentiate between superficially similar cases by applying context-sensitive reasoning, a capability that traditional single-agent models often lack. This is particularly critical in the regulatory space, where the same statutory provision may be interpreted differently depending on factual nuances, industry-specific norms, or jurisdictional precedents.

The category-specific performance analysis provides further clarity on the strengths of this approach. The exceptional improvement in Clear Compliance scenarios underscores the value of embedding domain-specific business context analysis directly into the reasoning process. By contrast, the more modest gains in Consent-Based Compliance cases suggest that some legal determinations remain inherently dependent on the subtleties of intent, documentation, and historical precedent—factors that cannot be fully captured by surface-level data analysis alone.

From a strategic perspective, these results hint at an emerging design principle for high-performance legal AI: instead of pursuing a single, universally competent model, it may be more effective to build a coalition of specialized reasoning modules, each optimized for a narrow domain

Table 1: Overall Performance Comparison

Approach	Accuracy	Precision	Recall	F1-Score
Single Agent	0.510	0.741	0.510	0.471
Multi-Agent	0.720	0.789	0.720	0.725
Improvement (% Change)	+0.210 (+41.2%)	+0.048 (+6.5%)	+0.210 (+41.2%)	+0.255 (+54.1%)

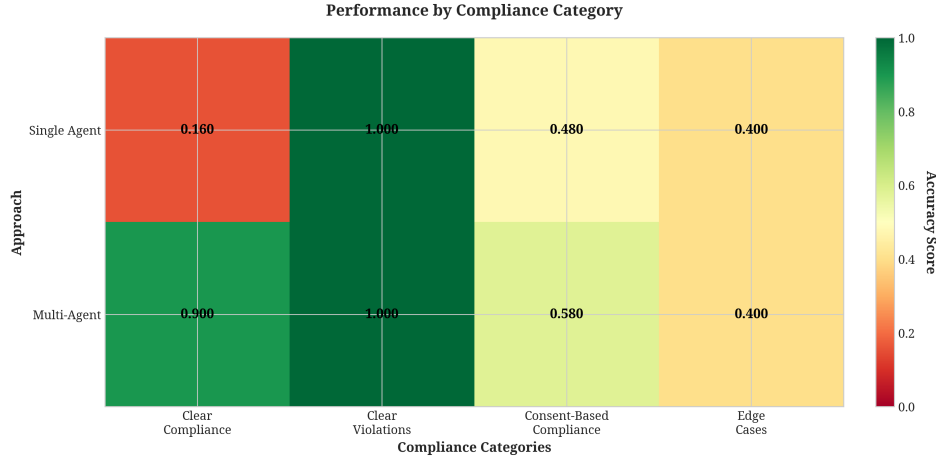


Figure 2: Performance Heatmap by Compliance Category. The visualization clearly shows the multi-agent system’s superior performance, particularly in Clear Compliance scenarios.

of expertise and coordinated through a structured decision-making framework. Such systems are not only more interpretable, but also more resilient to the weaknesses of any individual reasoning pathway.

7.2 Architectural Insights

The success of our multi-agent architecture can be attributed to several key design decisions:

1. **Specialization:** Each agent focuses on a specific aspect of legal analysis, allowing for deeper expertise development and more targeted reasoning capabilities.
2. **Coordination:** The coordinator agent synthesizes diverse perspectives while managing uncertainty, ensuring that decisions reflect a balanced integration of viewpoints rather than a single dominant interpretation.
3. **Confidence Integration:** The system incorporates calibrated confidence scores from specialized agents, enabling it to weigh contributions proportionally and make more informed, risk-adjusted final decisions.

7.3 Limitations and Future Work

Several limitations warrant discussion:

- **Edge Case Performance:** Neither approach shows improvement on genuinely ambiguous cases, suggesting fundamental challenges in legal uncertainty handling.
- **Computational Cost:** The 6.67x processing overhead may limit practical deployment in time-sensitive applications.

- **Evaluation Scope:** Our evaluation focuses on APPI Article 16; generalization to other regulations requires further investigation.

Building upon the identified limitations, future work will focus on enhancing the system’s capabilities and practical applicability. A primary objective is to address the fundamental challenge of handling legal ambiguity. To move beyond the current performance ceiling on genuinely ambiguous cases, future iterations will integrate comprehensive legal precedent and case law databases. This will enable the AI to interpret statutes not merely as abstract rules, but in the context of their real-world application by courts, thereby fostering a more nuanced and sophisticated approach to legal reasoning in complex scenarios.

Furthermore, to broaden the system’s utility beyond its current scope, we will explore dynamic agent specialization. Instead of being statically assigned to a single aspect of legal analysis, agents could adapt their expertise based on the specific regulatory domain in question, such as GDPR in Europe or CCPA in California. This dynamic specialization would not only test the model’s generalizability but also pave the way for tackling complex multi-jurisdictional compliance scenarios, a critical need for global enterprises.

Finally, to mitigate the significant computational overhead that currently limits practical deployment, we will investigate hybrid approaches designed to balance accuracy and efficiency. For instance, the system could employ a simpler, faster model for an initial screening of straightforward cases, reserving the resource-intensive multi-agent architec-

Table 2: Performance by Compliance Category

Category	Single Agent	Multi-Agent	Improvement	Sample Count
Clear Compliance	0.160	0.900	+0.740	50
Clear Violations	1.000	1.000	+0.000	50
Consent-Based Compliance	0.480	0.580	+0.100	50
Edge Cases	0.400	0.400	+0.000	50

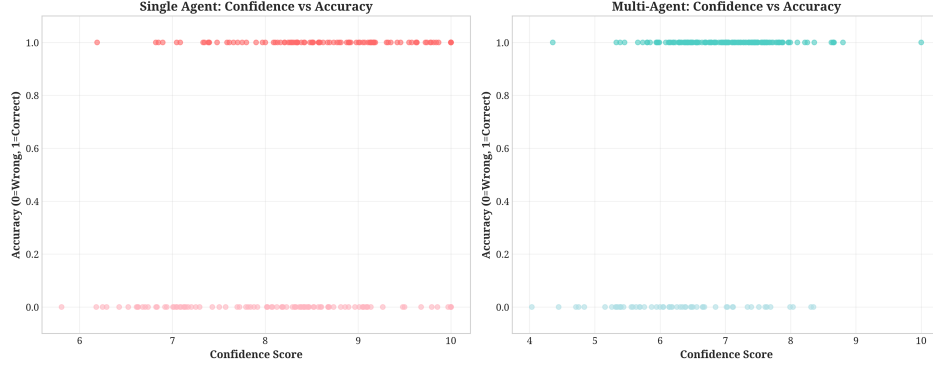


Figure 3: Confidence vs Accuracy Analysis for Single Agent and Multi-Agent Systems. The multi-agent system shows better confidence calibration with more reliable confidence scores.

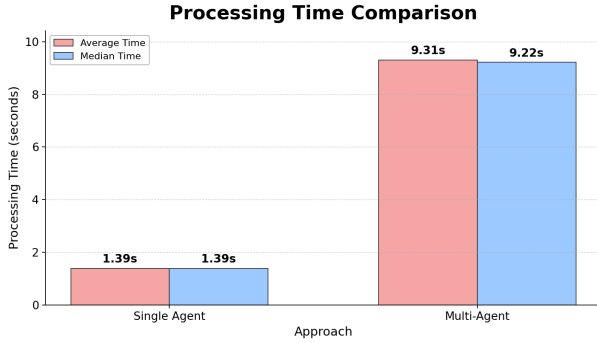


Figure 4: Processing Time Comparison showing the computational overhead of the multi-agent approach. While the multi-agent system requires 6.67x more processing time, this may be acceptable for high-stakes compliance decisions.

ture for situations that genuinely require deep, multifaceted analysis. Such a hybrid model would make the system viable for time-sensitive, real-world applications without substantially compromising the accuracy gains our research has demonstrated.

8 Conclusion

This paper presents a novel multi-agent legal verifier system for data transfer planning compliance, demonstrating significant improvements over single-agent approaches. Our experimental evaluation across 200 APPI Article 16 compliance scenarios shows a 21.0 percentage point accuracy improvement, with particularly strong performance in recognizing legitimate business uses of personal data.

The success of our approach validates the hypothesis that legal compliance checking benefits from specialized multi-agent architectures that decompose complex legal reasoning into manageable sub-problems. The coordination mechanism effectively synthesizes diverse analytical perspectives while maintaining appropriate uncertainty handling.

These results have important implications for the development of trustworthy AI systems operating under regulatory constraints. As privacy regulations continue to evolve and proliferate globally, automated compliance verification systems will become increasingly critical for AI deployment in data-sensitive applications.

Our work contributes to the growing intersection of AI and law by providing both a practical framework for automated compliance checking and empirical evidence supporting multi-agent approaches for complex legal reasoning tasks. The open questions raised by our edge case analysis point toward important directions for future research in legal AI systems.

References

- Alechina, N.; Halpern, J. Y.; Kash, I. A.; and Logan, B. 2016. Decentralised norm monitoring in open multi-agent systems. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, 1399–1400.
- Alur, R.; Courcoubetis, C.; and Dill, D. 1990. Model-checking for real-time systems. In *Proceedings of the Fifth Annual IEEE Symposium on Logic in Computer Science*, 414–425. IEEE.
- Amor, R., and Dimyadi, J. 2021. The promise of automated

- compliance checking. *Developments in the built environment* 5:100039.
- Arias, J.; Moreno-Rebato, M.; Rodriguez-García, J. A.; and Ossowski, S. 2024. Automated legal reasoning with discretion to act using s(LAW). *Artificial Intelligence and Law* 32(4):1141–1164.
- Burgemeestre, B.; Hulstijn, J.; and Tan, Y.-H. 2011. Value-based argumentation for justifying compliance. *Artificial Intelligence and Law* 19(2):149.
- Cerqueira, J. A. S.; Agbese, M.; Rousi, R.; Xi, N.; Hamari, J.; and Abrahamsson, P. 2024. Can we trust ai agents? a case study of an LLM-based multi-agent system for ethical ai. In *arXiv preprint arXiv:2411.08881*.
- Cliffe, O.; De Vos, M.; and Padget, J. 2006. Answer set programming for representing and reasoning about virtual institutions. In *International Workshop on Computational Logic in Multi-Agent Systems*, 60–79. Springer.
- Governatori, G., and Terenziani, P. 2007. Temporal extensions to defeasible logic. In *Australasian Joint Conference on Artificial Intelligence*, 476–485. Springer.
- Hayashi, H.; Mitsikas, T.; Taheri, Y. S.; Tsushima, K.; Schäfermeier, R.; Bourgne, G.; Ganascia, J.-G.; Paschke, A.; and Satoh, K. 2023. Multi-agent online planning architecture for real-time compliance. In *17th International Rule Challenge and 7th Doctoral Consortium@ RuleML+ RR 2023*.
- Javed, K., and Li, J. 2024. Artificial intelligence in judicial adjudication: Semantic biasness classification and identification in legal judgement (SBCILJ). *Heliyon* 10(9).
- King, T. C.; De Vos, M.; Dignum, V.; Jonker, C. M.; Li, T.; Padget, J.; and van Riemsdijk, M. B. 2017. Automated multi-level governance compliance checking. *Autonomous Agents and Multi-Agent Systems* 31(6):1283–1343.
- Liu, Q., and Liu, Y. 2018. Multi-agent epistemic planning with common knowledge. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 1912–1920.
- Mitrou, L. 2018. Data protection, artificial intelligence and cognitive services: is the general data protection regulation (gdpr) ‘artificial intelligence-proof’? *Artificial Intelligence and Cognitive Services: Is the General Data Protection Regulation (GDPR) ‘Artificial Intelligence-Proof’*.
- Nguyen, H. T., and Satoh, K. 2024. Krag framework for enhancing LLMs in the legal domain. In *Proceedings of the First International Workshop on Next-Generation Language Models for Knowledge Representation and Reasoning (NeLaMKRR 2024)*.
- Nguyen, H.-T.; Toni, F.; Stathis, K.; and Satoh, K. 2023. Beyond logic programming for legal reasoning. *arXiv preprint arXiv:2306.16632*.
- Nguyen, H. T.; Fungwacharakorn, W.; Zin, M. M.; Goebel, R.; Toni, F.; Stathis, K.; and Satoh, K. 2025. LLMs for legal reasoning: A unified framework and future perspectives. *Computer Law & Security Review* 58:106165.
- Nute, D. 1994. Defeasible logic. In *Handbook of Logic in Artificial Intelligence and Logic Programming*. Oxford University Press.
- Nute, D. 1997. *Defeasible deontic logic*, volume 263. Springer Science & Business Media.
- Padget, J.; ElDeen Elakehal, E.; Li, T.; and De Vos, M. 2016. InstAL: an institutional action language. In *Social coordination frameworks for social technical systems*. Springer. 101–124.
- Palmirani, M.; Governatori, G.; Rotolo, A.; Tabet, S.; Boley, H.; and Paschke, A. 2011. LegalRuleML: XML-based rules and norms. In *International Workshop on Rules and Rule Markup Languages for the Semantic Web*, 298–312. Springer.
- Prakken, H., and Sartor, G. 2015. Law and logic: A review from an argumentation perspective. *Artificial intelligence* 227:214–245.
- Robaldo, L.; Batsakis, S.; Calegari, R.; Calimeri, F.; Fujita, M.; Governatori, G.; Morelli, M. C.; Pacenza, F.; Pisano, G.; Satoh, K.; et al. 2024. Compliance checking on first-order knowledge with conflicting and compensatory norms: a comparison among currently available technologies. *Artificial Intelligence and Law* 32(2):505–555.
- Sadiq, S., and Governatori, G. 2014. Managing regulatory compliance in business processes. In *Handbook on business process management 2: Strategic alignment, governance, people and culture*. Springer. 265–288.
- Satoh, K.; Asai, K.; Kogawa, T.; Kubota, M.; Nakamura, M.; Nishigai, Y.; Shirakawa, K.; and Takano, C. 2010. Proleg: an implementation of the presupposed ultimate fact theory of japanese civil code by prolog technology. In *JSAI international symposium on artificial intelligence*, 153–164. Springer.
- Steenhuisen, J. R.; Witteveen, C.; ter Mors, A. W.; and Valk, J. M. 2006. Framework and complexity results for coordinating non-cooperative planning agents. In *German Conference on Multiagent System Technologies*, 98–109. Springer.
- Von Wright, G. H. 1951. Deontic logic. *Mind* 60(237):1–15.

HA-GNN: Learning Acyclic Hierarchies in Scientific Text with Hierarchy-Aware Graph Neural Networks

Devvrat Joshi and Islem Rekik*

BASIRA Lab, Imperial-X (I-X) and Department of Computing, Imperial College London, London, United Kingdom

Abstract

Extracting latent hierarchies from unstructured scientific text literature remains a fundamental challenge in computational knowledge discovery. However, state-of-the-art relation extraction models are limited to local intra-sentential reasoning and lack inductive biases that promote global structural coherence. This critical limitation prevents them from enforcing fundamental axioms of hierarchies, such as acyclicity, often resulting in logically inconsistent knowledge graphs that are unsuitable for downstream reasoning. To address this, we introduce the Hierarchy-Aware Graph Neural Network (HA-GNN), a framework that reframes relation extraction as a problem of joint structure discovery and representation learning. HA-GNN integrates two key innovations: (1) a probabilistic message-passing scheme that learns robust entity representations over an uncertain graph structure, and (2) a differentiable acyclicity loss that embeds this core hierarchical constraint directly into the optimization objective. This novel regularizer guides the model to learn a globally consistent directed acyclic graph. We evaluated our algorithm on two scientific literature datasets, SciER and SciERC, and demonstrated that HA-GNN improves the Rel+ F1 metric for the hierarchical *Part-Of* relation by 15-18% absolute points over state-of-the-art baselines. Our HA-GNN code is available at <https://github.com/basiralab/HA-GNN>.

1 Introduction

A fundamental challenge in computational knowledge discovery originates from the inherent disparity between the structured logic of scientific principles and their unstructured representation in scholarly text (Verma et al. 2023). As a consequence of this *epistemological gap*, the organized, hierarchical frameworks of scientific fields remain latent and computationally intractable within the vast corpus of literature. The primary objective of large-scale relation extraction is therefore not simply to index information, but to reconstruct this latent logic into a high-fidelity, structured representation (Magnusson and Friedman 2021), as conceptually illustrated in Figure 1. The success of this task depends on creating knowledge graphs that are structurally coherent and capable of representing complex scientific reasoning.

This demand for structural coherence directs attention to the foundational pillars of scientific organization: compositional (*Part-Of*) and taxonomic (*Subclass-Of*) relations. Unlike simpler associative links, these relational types are not only descriptive but represent the architectural principles that encode the logic of a domain (Vogt, Kuhn, and Hoehndorf 2024). Compositional relations specify how complex systems are assembled from constituent parts, while taxonomic relations establish the classification structures that group specific concepts into broader categories. The fidelity of the entire knowledge graph, its capacity for inference, logical validation, and ultimately the facilitation of new discovery, depends critically on the precise extraction of these specific hierarchical links (Zhang et al. 2018).

Current relation extraction methods, while powerful, reveal a fundamental architectural flaw when tasked with building knowledge hierarchies. State-of-the-art (SOTA) approaches often rely on fine-tuning large, pre-trained encoder models like SciBERT (Beltagy, Lo, and Cohan 2019). These models excel at learning rich, contextual representations from local sentence structure. For instance, models like PL-Marker (Ye et al. 2022) achieve SOTA results by inserting entity markers into text and processing the sequence, but this very design confines their reasoning to a narrow, intra-sentential window. They lack any mechanism to enforce global structural property such as acyclicity. *This architectural blind spot means they can easily predict a concept to be its own ancestor, producing a logically incoherent knowledge graph and severely limiting its utility for any downstream reasoning.*

This limitation remains evident even in advanced frameworks such as HGERE (Yan et al. 2023), which models higher order interactions through a hypergraph representation. Its relation-relation edges (for example, *sibling* and *grand-parent*) are derived from syntactic dependency parsing. While this technique is effective at capturing grammatical structure, it is fundamentally inadequate for inferring the semantic axioms that define real world hierarchical relations such as *Part-Of*. As a result, HGERE cannot enforce essential ontological constraints, including acyclicity. This syntactic orientation is further reinforced

*Corresponding author: i.rekik@imperial.ac.uk, <http://basira-lab.com>, GitHub: <https://github.com/basiralab/HA-GNN>

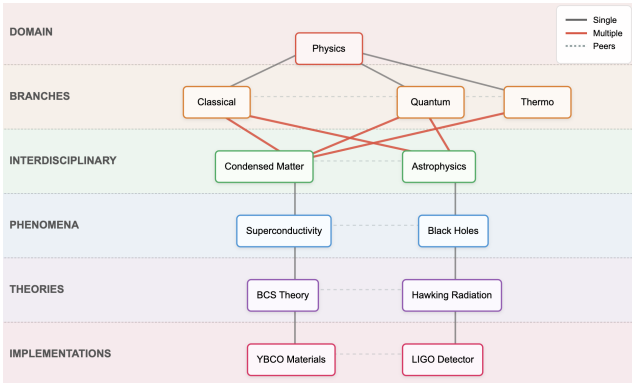


Figure 1: A conceptual illustration of the latent scientific hierarchy in Physics, highlighting how compositional and peer relations form a Directed Acyclic Graph (DAG) by allowing concepts to have multiple parents and children.

by its reliance on a span pruning mechanism to generate candidate entities, which confines the model to refining a pre-specified local graph. Collectively, these design choices position HGERE as a system of local syntactic refinement rather than global semantic induction, which is the requirement of our task.

The core challenge, therefore, extends beyond the simple classification of relations and lies in the discovery of a globally consistent latent hierarchy. To address this limitation, we propose the *Hierarchy-Aware Graph Neural Network (HA-GNN)*, which reframes relation extraction as a structure discovery problem. In this paradigm, the model simultaneously learns structure-aware entity representations and the relational graph itself.

HA-GNN overcomes the limitations of prior methods through two central architectural innovations. The first is *probabilistic message passing* over an uncertain graph, which allows the model to acquire robust entity representations while the relational structure is being inferred. The second, and most critical, is a *differentiable acyclicity loss*. This structural regularizer enforces global consistency by embedding the axiom that a valid hierarchy cannot contain cycles directly into the optimization. This novel constraint ensures the validity of the discovered hierarchy and leads to new state-of-the-art performance on the SciERC (Luan et al. 2018) and SciER (Zhang et al. 2024) datasets, improving the Rel+ F1 score on hierarchical relations by over 15-18 absolute points.

2 Related Work

The methodology for Relation Extraction (RE) has evolved significantly, progressing from early pipeline systems (Zelenko, Aone, and Richardella 2003) and joint modeling frameworks like (Miwa and Bansal 2016) and (Katiyar and Cardie 2017) to the current state-of-the-art dominated by Pre-trained Language Models (PLMs) (Devlin et al.

2019). Architectures leveraging powerful contextualized embeddings (Ye et al. 2022) and even higher-order inference via hypergraphs (Yan et al. 2023) have pushed performance boundaries by adeptly modeling local context and complex inter-entity dependencies. Despite these advances, a fundamental limitation persists across all prior work: a lack of inductive bias for global structural consistency. Existing models are optimized for local, sentence level reasoning and cannot enforce fundamental axioms, such as acyclicity, that are essential for inducing a valid semantic hierarchy from unstructured scientific text.

Our work addresses this specific, unaddressed gap. While research in knowledge graph completion has explored logical rule enforcement, such methods typically operate on a pre-existing graph structure (Guo et al. 2018). Our approach is distinct in that it learns the relational graph from raw text while simultaneously enforcing a core axiom of hierarchies through a differentiable loss function. This allows the model to learn not just the relations, but the globally consistent structure of the hierarchy itself. *To the best of our knowledge, this work is the first to jointly leverage hierarchical probabilistic message passing and an explicit acyclicity constraint within a single, end-to-end framework for structure discovery and precise hierarchical relation extraction from unstructured scientific text.*

3 The HA-GNN Architecture

Our model is designed to discover a latent hierarchical graph from a corpus by *unifying structure prediction and representation learning*. It consists of three main components: a Latent Relation Predictor to parameterize the unknown graph, a multi-channel Probabilistic Message Passing model to learn node embeddings on this uncertain structure, and a Differentiable Hierarchy Loss that enforces acyclicity. For the visual understanding of our approach, refer Figure 2.

3.1 Latent Relation Predictor

We model the unknown structure as a learnable function, treating the existence and type of a relation between any two concepts as a probabilistic variable. We define a finite set of possible relation types, $\mathcal{R} = \{\text{parent-of, peer-of, no-edge}\}$. For each pair of concept nodes (u, v) , we obtain their initial embeddings $(\mathbf{h}_u, \mathbf{h}_v)$ from SciBERT. These embeddings are concatenated and passed through a Multi-Layer Perceptron (MLP), which outputs a probability distribution over the set of possible relation types:

$$P_{uv} = \text{softmax}(\text{MLP}([\mathbf{h}_u || \mathbf{h}_v])) \quad (1)$$

where $||$ denotes vector concatenation. The output P_{uv} is a vector of probabilities $[P_{uv}^{\text{parent}}, P_{uv}^{\text{peer}}, P_{uv}^{\text{no-edge}}]$. This allows the model’s belief about the graph to be updated dynamically as the node embeddings improve.

3.2 Probabilistic Message Passing

Traditional GNNs operate on a fixed adjacency matrix. Since our graph is probabilistic, message passing must be “soft” and weighted by the model’s current relational beliefs.

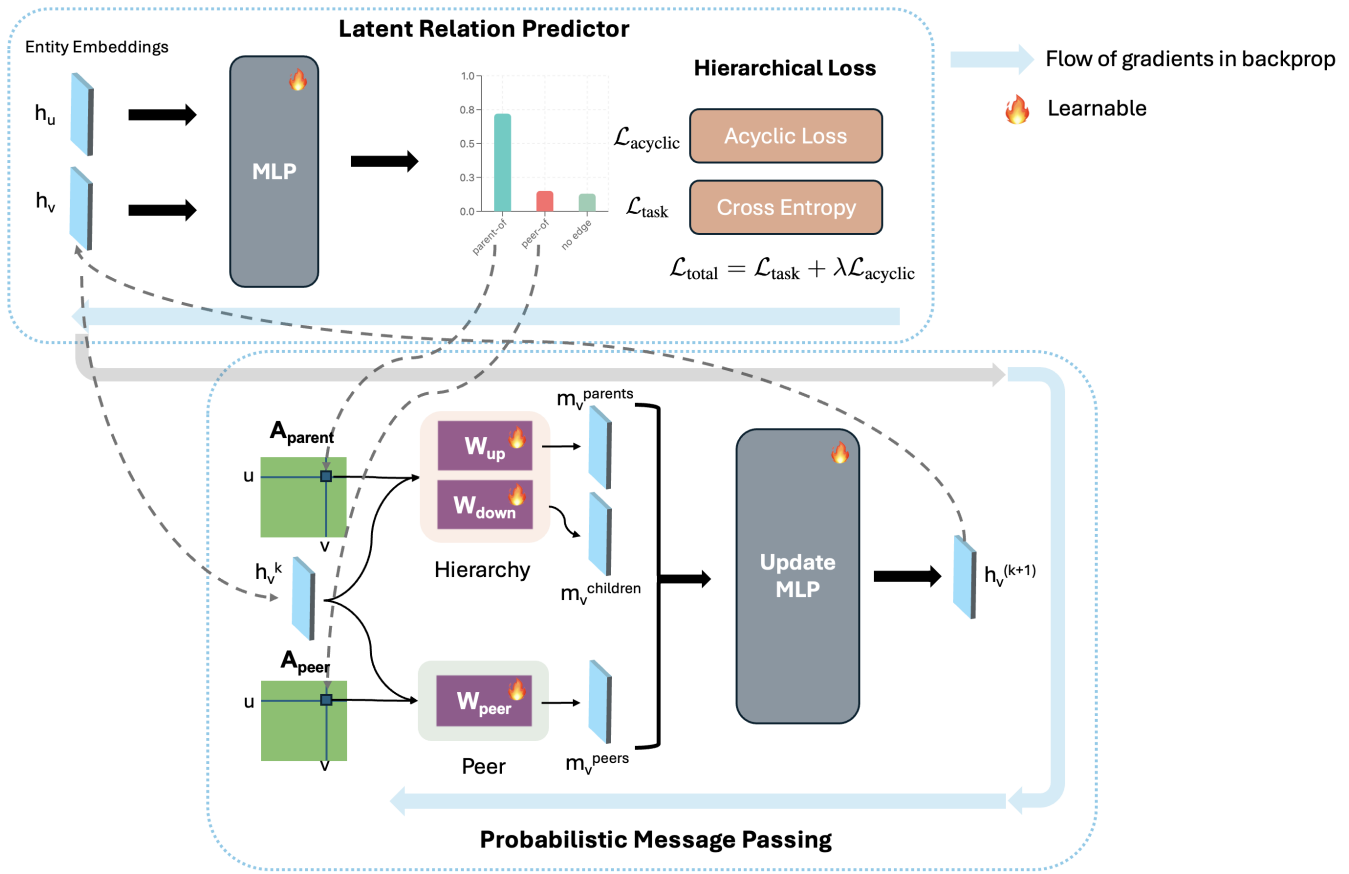


Figure 2: Our HA-GNN architecture with latent relation predictor, probabilistic message passing and acyclic loss.

We further refine this by separating the message passing into distinct channels based on the type of hierarchical relationship, allowing the model to distinguish between information flowing up, down, and across the hierarchy. For a given node v , its updated embedding is a function of its previous state and messages aggregated from three distinct contexts:

1. **Aggregate Upstream (Parental) Context:** Collect information from all potential parents of v .

$$\mathbf{m}_v^{\text{parents}} = \sum_{u \in V} P_{uv}^{\text{parent}} \cdot (\mathbf{W}_{\text{up}} \mathbf{h}_u) \quad (2)$$

2. **Aggregate Downstream (Child) Context:** Collect information from all potential children of v .

$$\mathbf{m}_v^{\text{children}} = \sum_{u \in V} P_{vu}^{\text{parent}} \cdot (\mathbf{W}_{\text{down}} \mathbf{h}_u) \quad (3)$$

3. **Aggregate Peer Context:** Collect information from all potential peers of v .

$$\mathbf{m}_v^{\text{peers}} = \sum_{u \in V} P_{uv}^{\text{peer}} \cdot (\mathbf{W}_{\text{peer}} \mathbf{h}_u) \quad (4)$$

where \mathbf{W}_{up} , \mathbf{W}_{down} , and \mathbf{W}_{peer} are learnable weight matrices for each relational channel. The final updated embedding for node v at iteration $k + 1$ is computed by combining

these aggregated messages with its previous state $\mathbf{h}_v^{(k)}$:

$$\mathbf{h}_v^{(k+1)} = \text{UpdateMLP}([\mathbf{h}_v^{(k)} || \mathbf{m}_v^{\text{parents}} || \mathbf{m}_v^{\text{children}} || \mathbf{m}_v^{\text{peers}}]) \quad (5)$$

This multi-channel approach provides a natural learning curriculum. Early in training, when probabilities are diffuse, updates are small and averaged. As confidence grows, strong messages along high-probability edges dominate, allowing the model to learn highly structure-aware representations.

3.3 Differentiable Hierarchy Loss

To guide the model towards learning a valid hierarchy, we introduce a custom loss function that encodes the fundamental axiom of acyclicity directly into the training objective. Local relation classification losses are insufficient as they cannot enforce global structural consistency. Let A_{parent} be the matrix of predicted parent-of probabilities, P_{uv}^{parent} , used in message passing. The total loss is:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda \mathcal{L}_{\text{acyclic}} \quad (6)$$

where $\mathcal{L}_{\text{task}}$ is a standard cross-entropy loss on the known labels, and $\mathcal{L}_{\text{acyclic}}$ is our structural regularizer.

Acyclicity Loss ($\mathcal{L}_{\text{acyclic}}$): A hierarchy, by definition, cannot contain cycles. We use a differentiable matrix-based for-

mulation to enforce this hard constraint. A well-known result from graph theory states that a graph is acyclic if and only if $\text{tr}(e^A) = N$, where A is the adjacency matrix and N is the number of nodes. We adapt our own variant of the acyclic loss in (Zheng et al. 2018) to our probabilistic setting as follows:

$$\mathcal{L}_{\text{acyclic}} = \text{tr}(e^{A_{\text{parent}}}) - N \quad (7)$$

This loss is differentiable with respect to the probabilities in A_{parent} and backpropagates a strong signal that penalizes any configuration of probabilities that would lead to cycles, forcing the model to learn a Directed Acyclic Graph (DAG) structure.

We finally predict each relation within the dataset using the enhanced embeddings from HA-GNN through another MLP which is similar to all other methods.

3.4 Proof of Acyclicity Loss

This proof connects the properties of a graph’s adjacency matrix to the provided loss function. Let A be the adjacency matrix of a directed graph with N nodes. For time complexity analysis, refer appendix .

Definitions and Proof:

1. **Paths and Matrix Powers:** A fundamental property of an adjacency matrix A is that the entry $(A^k)_{ij}$ in the k -th power of the matrix counts the number of distinct paths of length k from node i to node j .
2. **Cycles and the Trace:** A cycle of length k is a path from a node back to itself. Therefore, the number of cycles of length k starting and ending at node i is given by the diagonal element $(A^k)_{ii}$. Summing all the diagonal elements gives the trace of the matrix, $\text{tr}(A^k)$, which counts the total number of cycles of length k in the entire graph.
3. **The Matrix Exponential (Güttel and Nakatsukasa 2016):** The matrix exponential e^A is defined by its Taylor series expansion (Can be efficiently calculated using ‘torch.linalg.matrix_exp’ function):

$$e^A = I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \dots = \sum_{k=0}^{\infty} \frac{A^k}{k!}$$

Taking the trace of the matrix exponential sums the traces of all powers of A :

$$\text{tr}(e^A) = \sum_{k=0}^{\infty} \frac{\text{tr}(A^k)}{k!} = \text{tr}(I) + \text{tr}(A) + \frac{\text{tr}(A^2)}{2!} + \dots$$

4. **The Acyclicity Condition:** A graph is a **Directed Acyclic Graph (DAG)** if and only if it has no cycles of any length $k \geq 1$. This means $\text{tr}(A^k) = 0$ for all $k \geq 1$. In this case, the trace of the matrix exponential simplifies to:

$$\text{tr}(e^A) = \text{tr}(I) = N$$

where N is the number of nodes (the dimension of the identity matrix I).

5. **The Loss Function:** The loss function $\mathcal{L}_{\text{acyclic}} = \text{tr}(e^{A_{\text{parent}}}) - N$ is designed to be zero only when the graph is acyclic. By minimizing this loss, the optimizer is forced to drive the predicted *parent-of* probability matrix A_{parent} towards a state where the graph it represents has no cycles, thus enforcing the desired acyclic structure.

Time Complexity Analysis of Acyclicity Loss The primary computational cost of the acyclicity loss comes from calculating the matrix exponential. To keep this process efficient, the model uses Pade’s approximation along with a k-nearest neighbors approach. This method adds a time complexity of only $O(nk^2)$, where n is the total number of entities and k is the number of neighbors (set to a constant 20 in the experiments). By constraining the calculation to top 20 most probable neighbours for each entity, this approach avoids the more expensive computations typically required for matrix exponentials on dense graphs, making the structural regularizer scalable and practical for document-level analysis.

4 Experiments and Results

4.1 Datasets and Baselines

We conduct experiments on two challenging scientific information extraction benchmarks:

- **SciERC (Luan et al. 2018):** Contains abstracts from AI research papers with annotations for entities, relations, and coreference. It includes *Part-Of* and *Subclass-Of* relations.
- **SciER (Zhang et al. 2024):** A larger-scale successor to SciERC, with more abstracts and a more diverse set of relations, providing a rigorous testbed.

Table 1: Relation Distribution Statistics (SciER dataset)

Relation Type	Count	Percentage (%)
No-Relation	50,589	85.2643
Part-Of	1,865	3.1433
Used-For	2,398	4.0417
Evaluated-With	863	1.4545
Synonym-Of	880	1.4832
Compare-With	875	1.4748
SubClass-Of	697	1.1747
Benchmark-For	551	0.9287
SubTask-Of	210	0.3539
Trained-With	404	0.6809

We compare our HA-GNN against two state-of-the-art baselines:

1. **PL-Marker (Ye et al. 2022):** A powerful baseline that inserts entity markers into text and processes the sequence through a pre-trained language model. It achieves strong results for relations expressed within a single sentence but is inherently local in scope and lacks mechanisms to enforce global structural constraints.
2. **HGERE (Yan et al. 2023):** An advanced model that uses a hypergraph to model higher-order interactions. However, its architecture is grounded in syntactic dependency

parsing and is designed to refine a pre-existing local graph of candidate entities, rather than discover a latent, globally consistent semantic hierarchy from unstructured text.

4.2 Evaluation Metric

[h] Rel+ F1 (Zhong and Chen 2021) is the strict evaluation metric for end-to-end relation extraction, which requires the model to correctly predict the boundaries and types of the subject entity and the object entity, as well as the entity relation.

4.3 Dataset Analysis

To validate the foundational premise of our work, we conducted an empirical analysis of the dataset’s hierarchical structure. As detailed in Table 1, compositional (*Part-Of*) relations in SciER dataset are prevalent throughout the corpus. Critically, this analysis confirmed that the subgraph formed exclusively by these *Part-Of* connections is a directed acyclic graph (DAG), with no cycles detected across the entire dataset. This finding empirically justifies our model’s explicit acyclicity constraint. A further topological analysis of this DAG structure revealed a maximum depth of 9, underscoring the non-trivial hierarchical complexity latent within the scientific text.

4.4 Experimental Setup

Our HA-GNN is implemented in PyTorch. We use SciBERT (Beltagy, Lo, and Cohan 2019) for initial embeddings. HA-GNN model consists of 3 probabilistic message passing layers. We train for 20 epochs using the AdamW optimizer with a learning rate of 2×10^{-5} and a linear warmup scheduler. The acyclicity loss weight λ is set to 0.5. We use early stopping based on the development set’s Macro-F1 score.

4.5 Results

As detailed in Table 2, our experiments on the SciERC and SciER benchmarks establish HA-GNN as the new state-of-the-art (SOTA), substantially outperforming previous SOTA like PL-Marker and HGERE. The performance gains are most pronounced for hierarchical relations, which are the primary focus of our work. For instance, on both datasets, our model delivered significant improvements for *Part-Of* and *Subclass-Of* relations, boosting F1 scores by a wide margin. Furthermore, our model with the acyclic loss correctly reconstructs the longest hierarchical chains, showcasing its superior ability to learn deep global structures (refer table 3). These results validate the core efficacy of our approach, demonstrating that jointly modeling structure discovery with an explicit acyclicity constraint is highly effective for relations that demand global consistency.

The outcomes also underscore a fundamental limitation in prior models. While baselines exhibit strong local, sentence-level reasoning capabilities, they struggle to capture the inductive biases necessary for discovering acyclic, taxonomic, and compositional structures. Their architectural design lacks a mechanism for enforcing global coherence, which is essential for this task. In contrast, HA-GNN’s consistent

superiority across both datasets and all hierarchical relation types highlights the robustness and generalizability of our method, confirming the critical role of global structural axioms in reconstructing latent scientific knowledge.

4.6 Ablation Study

To assess the contribution of the acyclicity loss $\mathcal{L}_{\text{acyclic}}$, we evaluate a variant of HA-GNN with this component removed (HA-GNN w/o $\mathcal{L}_{\text{acyclic}}$ in Table 2). Across all relation types and datasets, removing this constraint produces a consistent performance drop.

For example, *Part-Of* on SciERC decreases by over 11 points compared to the full model. Although the ablated variant still surpasses PL-Marker and HGERE, the reduction highlights the role of $\mathcal{L}_{\text{acyclic}}$ in enforcing structural coherence and enabling more accurate extraction of hierarchical relations. This demonstrates that while the probabilistic message-passing architecture is inherently strong, the acyclicity constraint is essential for achieving the highest performance.

4.7 Qualitative Error Analysis

Corrected Error: Preventing Hierarchical Cycles

A key strength of HA-GNN is its ability to enforce global acyclicity, correcting errors that local models often make. Consider a paper in the SciER dataset that discusses a new neural network architecture.

- Sentence 1: Our model contains a multi-head attention mechanism, which is a core part of the transformer block.
- Sentence 2: The overall performance of the transformer block is primarily defined by the efficiency of its multi-head attention mechanism.

A local, sentence-level model like PL-Marker correctly identifies the *Part-Of* relation from the first sentence: (multi-head attention mechanism, *Part-Of*, transformer block). However, the second sentence is ambiguous. Its phrasing (“defined by”) could lead the local model to incorrectly infer a reverse, cyclical relationship: (transformer block, *Part-Of*, multi-head attention mechanism). This creates a logical impossibility where the block is a part of its own component.

HA-GNN corrects this error. When the model learns the initial path (multi-head attention \rightarrow transformer block), the acyclicity loss ($\mathcal{L}_{\text{acyclic}}$) actively penalizes any subsequent prediction that would complete a cycle. The gradient from this loss function forces the model to discard the illogical reverse relation, ensuring the final knowledge graph remains a valid Directed Acyclic Graph (DAG).

Common Failure Mode: Long-Range Coreference Resolution

While powerful, HA-GNN can still fail when hierarchical relationships depend on resolving complex or long-range coreferences across a document. Imagine a scientific paper that introduces a complex system.

- Sentence 1: We propose a novel Graph-based Recommendation System (GRS). It uses a node embedding layer to

Models	SciERC			SciER		
	Part-Of	Subclass-Of	Overall	Part-Of	Subclass-Of	Overall
PL-Marker (Ye et al. 2022)	35.90	34.63	42.17	42.12	33.94	56.73
HGERE (Yan et al. 2023)	38.15	32.54	43.86	46.82	32.80	58.47
HA-GNN (ours)	56.40 _{+18.25}	44.24 _{+11.70}	53.19 _{+9.33}	61.76 _{+14.94}	42.93 _{+10.13}	65.38 _{+6.91}
HA-GNN w/o $\mathcal{L}_{acyclic}$	45.27	38.55	47.13	55.30	40.78	62.51

Table 2: Rel+ F1 scores (%) of different models on the SciERC and SciER datasets for two relation types (*Part-Of* and *Subclass-Of*). We also report the overall F1 scores across all relation types for both datasets. We have used the same entity prediction method for all the models to have fair comparison of their relation prediction mechanisms. Subscripts indicate the improvement over SOTA model HGERE. All values are rounded-off to two decimal points.

represent user preferences. Here, the model correctly extracts (node embedding layer, Part-Of, GRS).

- Sentence 2: To handle data sparsity, we enhanced the feature representation component with a variational autoencoder. This significantly improved the model’s accuracy on benchmark tasks.

In this case, the phrase feature representation component in the second paragraph is a coreference for the node embedding layer introduced in the first. The document provides no explicit link between them. HA-GNN, like other models, struggles to connect these two distinct mentions. Because it fails to resolve that they are the same entity, it cannot infer that the variational autoencoder is also a sub-component of the node embedding layer. This limitation arises because the model’s message-passing mechanism is more effective at capturing explicit, nearby relationships than inferring identity from distant, abstract descriptions.

Table 3: Analysis of the longest correctly predicted hierarchical chain on the SciER dataset. The length signifies the maximum depth of a multi-step *Part-Of* relationship that a model correctly identified. A greater length demonstrates a superior ability to capture deep, global hierarchical structures.

Model	Longest Correct Chain
PL-Marker (Ye et al. 2022)	3
HGERE (Yan et al. 2023)	3
HA-GNN w/o $\mathcal{L}_{acyclic}$	4
HA-GNN (ours)	6

5 Conclusion & Future Work

This work introduced HA-GNN, a hierarchy-aware graph neural network that unifies probabilistic hierarchical structure discovery with a differentiable acyclicity constraint, enabling accurate extraction of hierarchical scientific relations from unstructured text. By embedding the axiom of acyclicity directly into the learning objective, HA-GNN consistently outperforms state-of-the-art baselines on both the SciERC and SciER benchmarks, delivering substantial gains for *Part-Of* and *Subclass-Of* relations. Our analysis demonstrates that this improvement is not merely a by-product of stronger local reasoning, but arises from the model’s ability to enforce global structural coherence throughout the learning process.

While the present study focuses on taxonomic and compositional relations in scientific literature, the proposed framework opens several promising directions for future research. Extending the architecture to jointly model multiple ontological constraints such as transitivity and mutual exclusivity could further improve logical consistency. Incorporating cross-document reasoning and temporal dynamics would also enable richer scientific knowledge graphs that evolve over time. Finally, applying HA-GNN to other domains with inherently hierarchical structure, such as biomedical ontologies, may reveal broader applicability and inspire new domain-specific adaptations.

References

- Beltagy, I.; Lo, K.; and Cohan, A. 2019. SciBERT: A pre-trained language model for scientific text. In Inui, K.; Jiang, J.; Ng, V.; and Wan, X., eds., *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 3615–3620. Hong Kong, China: Association for Computational Linguistics.
- Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, J.; Doran, C.; and Solorio, T., eds., *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171–4186. Minneapolis, Minnesota: Association for Computational Linguistics.
- Guo, S.; Wang, Q.; Wang, L.; Wang, B.; and Guo, L. 2018. Knowledge graph embedding with iterative guidance from soft rules. *Proceedings of the AAAI Conference on Artificial Intelligence* 32(1).
- Güttel, S., and Nakatsukasa, Y. 2016. Scaled and squared subdiagonal padé approximation for the matrix exponential. *SIAM Journal on Matrix Analysis and Applications* 37(1):145–170.
- Katiyar, A., and Cardie, C. 2017. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. In Barzilay, R., and Kan, M.-Y., eds., *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 917–928. Vancouver, Canada: Association for Computational Linguistics.
- Luan, Y.; He, L.; Ostendorf, M.; and Hajishirzi, H. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In Riloff, E.; Chiang, D.; Hockenmaier, J.; and Tsujii, J., eds., *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3219–3232. Brussels, Belgium: Association for Computational Linguistics.
- Magnusson, I., and Friedman, S. 2021. Extracting fine-grained knowledge graphs of scientific claims: Dataset and transformer-based results. In Moens, M.-F.; Huang, X.; Specia, L.; and Yih, S. W.-t., eds., *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 4651–4658. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics.
- Miwa, M., and Bansal, M. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In Erk, K., and Smith, N. A., eds., *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1105–1116. Berlin, Germany: Association for Computational Linguistics.
- Verma, S.; Bhatia, R.; Harit, S.; and Batish, S. 2023. Scholarly knowledge graphs through structuring scholarly communication: a review. *Complex & Intelligent Systems* 9(1):1059–1095. Epub 09 August 2022.
- Vogt, L.; Kuhn, T.; and Hoehndorf, R. 2024. Semantic units: organizing knowledge graphs into semantically meaningful units of representation. *Journal of Biomedical Semantics* 15(1):7. Erratum in: *J Biomed Semantics*. 2024 Jun 6;15(1):10. doi: 10.1186/s13326-024-00313-2.
- Yan, Z.; Yang, S.; Liu, W.; and Tu, K. 2023. Joint entity and relation extraction with span pruning and hypergraph neural networks. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 7512–7526. Singapore: Association for Computational Linguistics.
- Ye, D.; Lin, Y.; Li, P.; and Sun, M. 2022. Packed levitated marker for entity and relation extraction. In Muresan, S.; Nakov, P.; and Villavicencio, A., eds., *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 4904–4917. Dublin, Ireland: Association for Computational Linguistics.
- Zelenko, D.; Aone, C.; and Richardella, A. 2003. Kernel methods for relation extraction. *Journal of machine learning research* 3(Feb):1083–1106.
- Zhang, Z.; Zhuang, F.; Qu, M.; Lin, F.; and He, Q. 2018. Knowledge graph embedding with hierarchical relation structure. In Riloff, E.; Chiang, D.; Hockenmaier, J.; and Tsujii, J., eds., *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3198–3207. Brussels, Belgium: Association for Computational Linguistics.
- Zhang, Q.; Chen, Z.; Pan, H.; Caragea, C.; Latecki, L. J.; and Dragut, E. 2024. SciER: An entity and relation extraction dataset for datasets, methods, and tasks in scientific documents. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 13083–13100. Miami, Florida, USA: Association for Computational Linguistics.
- Zheng, X.; Aragam, B.; Ravikumar, P. K.; and Xing, E. P. 2018. Dags with no tears: Continuous optimization for structure learning. In Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.
- Zhong, Z., and Chen, D. 2021. A frustratingly easy approach for entity and relation extraction. In Toutanova, K.; Rumshisky, A.; Zettlemoyer, L.; Hakkani-Tur, D.; Beltagy, I.; Bethard, S.; Cotterell, R.; Chakraborty, T.; and Zhou, Y., eds., *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 50–61. Online: Association for Computational Linguistics.

Which Neurons Nudge Moral Judgments? Neuron-Level Last-Token Steering in Large Language Models

Davide Liga^{*1}, Liuwen Yu¹, Réka Markovich¹

¹University of Luxembourg

{davide.liga, liuwen.yu, reka.markovich}@uni.lu

Abstract

We study how to steer a language model’s moral judgments at inference time by adding a tiny, contrastive perturbation to the *last-token* neural activation in late MLP layers (contrastive last-token steering). For each moral prompt, we construct a contrast direction by comparing its last-token activation to that of a minimally edited variant that implies a more permissive moral stance (e.g., “acceptable” rather than “wrong”). During generation, we add this vector at the last token; a single strength α controls how strongly and in which direction we push the model’s stance (permissive vs. restrictive).

Impact is measured as the change in a next-token logit margin between permissive and restrictive continuations. To avoid overclaiming, we calibrate a threshold τ on neutral controls (same layers, tempered strengths with $|\alpha| \leq 1$) and count success only when the shift exceeds τ in the expected direction. We also assess *specificity* by verifying that, on neutral control prompts, steered outputs exactly match unsteered baselines. Beyond component-level tests, we probe *neuron-level locality* by steering only the top- k contrastive neurons (ranked by last-token contrast) and confirming reversibility on our test set: $+\alpha$ produces the shift and $-\alpha$ reverses it. The method is training-free, uses standard forward hooks, and we report pilot results on Llama-3-8B-Instruct.

1 Introduction

Large language models (LLMs) now routinely engage in tasks that require moral judgment—choosing between conflicting values, resolving ethical dilemmas, and aligning with user-defined principles. When prompted with trolley-style scenarios or everyday social dilemmas, these models can articulate moral reasoning and produce graded or binary decisions. Yet while the outputs are often coherent, the internal mechanisms that steer these choices remain opaque. Which internal representations are responsible for a model’s moral stance, and can we *causally* intervene on them without retraining?

Terminology. We call a stance *permissive* when continuations such as “acceptable”/“justified” are preferred over “wrong”/“unacceptable”, and *restrictive* otherwise. *Specificity* means the intervention leaves unrelated outputs unchanged; we measure it as canonicalized exact-match of

steered vs. unsteered greedy generations on neutral controls. For example, the minimal pair “Lying to save lives is morally...” vs. “Lying to save face is morally...” implies opposite stances under small textual changes.

Our approach. We study these questions via *contrastive activation steering*. Rather than swapping whole activations, we compute a *direction of change* between a minimally different prompt pair that elicits opposite stances (“permissive” vs. “restrictive”). We then inject a small, controlled perturbation along this contrast direction at the *last input position* inside selected MLP blocks (primarily late layers) during the forward pass. The intervention is scaled by a single real parameter α (the *strength*); positive $+\alpha$ is aligned—by a brief sign probe—to push toward the stance predicted by the counterfactual prompt, while negative $-\alpha$ reverses the effect. We quantify causal impact using a calibrated *next-token logit margin* between mutually exclusive continuations corresponding to the two stances. For moral pairs we always steer the last token; for neutral controls we keep the same last-token position but use masking to check that outputs remain unchanged.

Evaluation frame. To avoid over-interpreting single runs, we adopt a compact but principled evaluation pipeline:

1. **Directional validity.** At a given α , does the decision shift Δ have the expected sign (toward permissive for $+\alpha$, toward restrictive for $-\alpha$) as predicted by the counterfactual pair?
2. **Calibrated decision threshold.** We estimate a threshold τ from neutral control prompts using the same layers and a tempered subset of strengths ($|\alpha| \leq 1$, i.e., $\alpha \in \{-1.0, -0.8, 0.8, 1.0\}$), and count a *success* only when $\Delta \cdot \text{sign expected}$ exceeds τ .
3. **Specificity.** We test that steering leaves unrelated outputs unchanged by comparing steered vs. unsteered generations on neutral controls (canonicalized exact match), reporting preservation and corruption rates.
4. **Dose-response and reversibility.** We sweep α over a symmetric grid (positive/negative). Neuron-level tests include explicit reversibility checks ($+\alpha$ vs. $-\alpha$). At the component level we report success across strengths; a formal monotonicity score is left for future work.
5. **Neuron-level locality.** Within late MLP layers, we select top- k neurons by last-token contrast magnitude and test

^{*}Corresponding author

whether small, sign-aligned subsets can reproduce (and reverse) the steering effect.

Contributions. We present a compact pipeline for causally steering moral stances in pretrained LLMs:

1. **Contrastive last-token steering.** A causal intervention that targets *contrastive*, *last-token* directions by modifying MLP outputs that feed the residual stream (we emphasize late layers but also test selected earlier layers). The method requires no fine-tuning and uses standard forward hooks during inference only.
2. **Calibrated logit-margin metric.** A decision-shift metric based on the change in logit margin between opposed completions, with a neutral-control-derived threshold for consistent comparisons across prompts, layers, and steering strengths.
3. **Neuron-level locality.** A neuron-level analysis in late MLPs showing that small, sign-aligned subsets (top- k by contrast) can flip stance on our test set and exhibit reversibility with $+\alpha/-\alpha$.

Together, these components provide a reproducible workflow for measuring and controlling moral decisions in LLMs using only inference-time interventions.

Scope. We evaluate on Meta-Llama-3-8B-Instruct and a set of moral minimal pairs; claims are about steering effects within this setting. Component-level dose-response is reported via strength sweeps; explicit monotonicity scoring is left as future work.

Paper organization. Section 2 contextualizes this work in the literature. Section 3 presents the contrastive last-token steering method. Section 4 details the experimental setup and reports results. Section 5 discusses limitations and future directions. Section 6 concludes.

2 Related Work

Moral behavior in language models. LLMs exhibit moral and social judgments that appear to reflect implicit values learned during pretraining. Prior work has evaluated these behaviors using curated benchmarks and ethical dilemmas. For example, Jiang et al. (2021) introduced Delphi to make ethical judgments across diverse scenarios, and Hendrycks et al. (2020) proposed ETHICS to assess dimensions such as justice, virtue, and utilitarianism. Abdulhai et al. (2023) analyze moral foundations in LLMs, while Madaan et al. (2023) explore self-refinement strategies that can influence responses to moral questions. Much of this literature is output-centered; the internal mechanisms producing these stances remain comparatively less characterized.

Mechanistic interpretability and causal interventions. Mechanistic interpretability aims to reverse-engineer computations inside transformers (Nanda et al., 2023), including work on superposition and feature sharing (Elhage et al., 2022) and on targeted weight edits such as ROME (Meng et al., 2022). A complementary line of research uses *activation patching*/causal tracing to test how swapping or modifying hidden states affects behavior. Our approach follows this causal-intervention paradigm but in a localized setting: we construct a *contrastive direction* from minimally different

prompt pairs and add a small, inference-time perturbation at the *last token* inside late MLP blocks, leaving model weights unchanged.

Probing vs. causal control. Linear probes and concept vectors (e.g., CAVs; Kim et al., 2018) indicate that a representation correlates with a concept, but correlation does not by itself imply that manipulating that representation *causes* behavior to change. Bias and social-attribute evaluations (e.g., Nangia et al., 2020) similarly diagnose tendencies without isolating mechanism. By contrast, we emphasize *causal* tests: we measure a calibrated change in a next-token logit margin when we add a small, sign-aligned perturbation to late-layer MLP activations. Beyond component-level tests, we examine *neuron-level locality* by steering only the top- k contrastive neurons and verifying reversibility with $+\alpha/-\alpha$.

Steering without fine-tuning. Inference-time steering spans prompt-based control, activation editing, and representation arithmetic. Our contribution fits within this space but differs in three respects: (i) we derive *contrastive*, *last-token* directions from minimally changed moral pairs (rather than global directions or weight edits), (ii) we *calibrate* a decision threshold on neutral controls to avoid overclaiming spurious flips, and (iii) we validate *locality and reversibility* by showing that small neuron subsets in late MLPs can reproduce and reverse the effect. This yields an efficient, reproducible pipeline for causal moral steering using standard forward hooks at inference time.

3 Method

We describe an interventional, inference-time procedure to steer a pretrained language model’s moral stance by adding small, targeted perturbations to late MLP activations. The procedure has four components: (i) a *decision signal* defined as a next-token logit margin between mutually exclusive moral continuations; (ii) *contrastive last-token steering* that injects a direction computed from minimally different prompt pairs; (iii) *calibration and specificity* using neutral controls; and (iv) a *neuron-level* variant that targets small subsets of units.

At a glance. For each moral pair we (a) build a last-token direction from the base vs. source contrast (or a PCA fallback—last-token, position-aware—if unavailable), (b) orient it via a one-shot *sign probe* and inject it into selected late MLP layers at strength α , and (c) score the change in the next-token margin against a calibrated threshold; we also (d) check specificity on neutral controls and (e) run a neuron-level variant targeting top- k units. We focus on last-token interventions because late-layer MLP features tend to consolidate decision signals; in practice our component-level run steers a fixed set of MLP layers spanning early through late blocks, while neuron-level tests focus on late layers where the effect is most localized. The steering strength α is not learned: we sweep a symmetric grid and select α by a calibrated success metric, with the threshold τ estimated from neutral controls using the same layers and tempered strengths. A compact overview of the pipeline is shown in Figure 1.

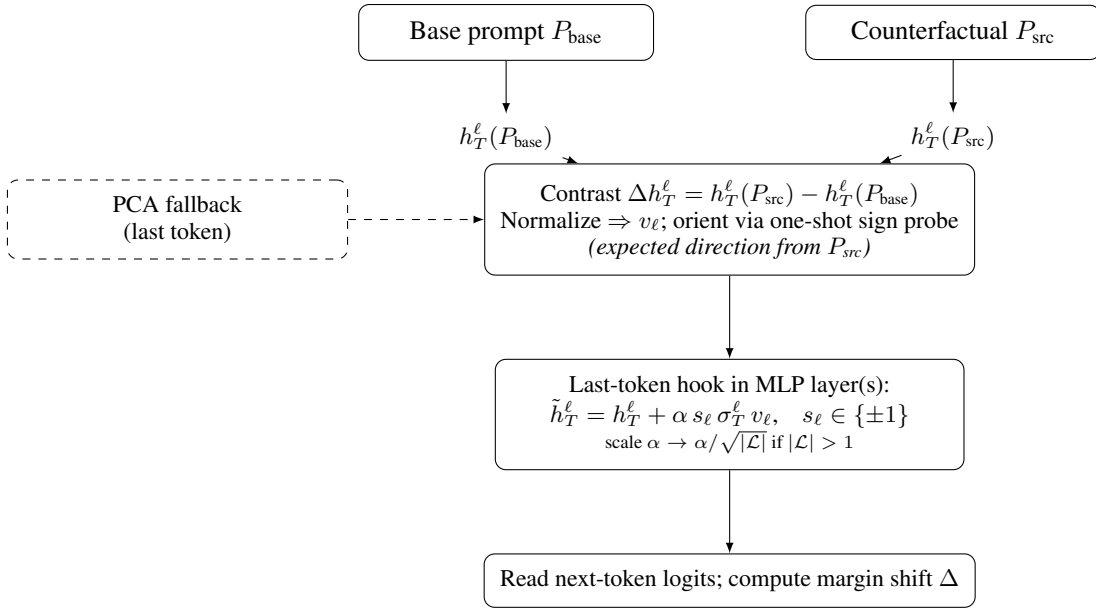


Figure 1: **Contrastive last-token steering pipeline (schematic)**. (1) Encode P_{base} and P_{src} ; take last-token MLP activations $h_T^\ell(\cdot)$. (2) Form the contrast Δh_T^ℓ and normalize to v_ℓ ; orient via a one-shot sign probe using the expected direction from the source completion. (3) During generation, hook the last token in layer(s) \mathcal{L} and apply $\tilde{h}_T^\ell = h_T^\ell + \alpha s_\ell \sigma_T^\ell v_\ell$ (scale $\alpha \rightarrow \alpha/\sqrt{|\mathcal{L}|}$ if $|\mathcal{L}| > 1$). (4) Read next-token logits and compute the margin shift Δ . Dashed arrow: optional PCA fallback for v_ℓ .

3.1 Task and Decision Signal

We work with *minimal pairs* of prompts ($P_{\text{base}}, P_{\text{src}}$) that differ by a small surface change but imply opposite stances (“permissive” vs. “restrictive”). Let x be the tokenized input for P_{base} , and let y_+ and y_- denote mutually exclusive next-token continuations aligned with “permissive” and “restrictive” readings, respectively.¹ Given next-token logits $\ell(\cdot)$, we define the *logit margin*

$$m = \ell(y_+) - \ell(y_-),$$

and the *decision shift* for an intervention as

$$\Delta = m_{\text{patched}} - m_{\text{base}}.$$

Positive Δ indicates movement toward a permissive decision.

3.2 Contrastive Last-Token Steering

Let $h^\ell(x) \in \mathbb{R}^{T \times H}$ be the MLP output at layer ℓ for input x (sequence length T , hidden size H). We steer only the *last token*. Define the last-token shorthand

$$h_T^\ell(P) \equiv h^\ell(P)_{T,:},$$

and form a per-pair *contrastive direction* by differencing the base and counterfactual prompts:

$$\begin{aligned} \Delta h_T^\ell &:= h_T^\ell(P_{\text{src}}) - h_T^\ell(P_{\text{base}}), \\ v_\ell &:= \frac{\Delta h_T^\ell}{\|\Delta h_T^\ell\|_2 + \varepsilon}, \end{aligned}$$

¹In practice, we pool a small set of stance tokens/phrases such as *right/acceptable/justified* vs. *wrong/unacceptable/unjustified*.

where ε is a small device/dtype-safe constant.

At inference time, inside a selected set of MLP layers \mathcal{L} , we add a tiny perturbation at the last position:

$$\tilde{h}_T^\ell = h_T^\ell + \alpha s_\ell \sigma_T^\ell v_\ell, \quad \ell \in \mathcal{L},$$

where $\alpha \in \mathbb{R}$ is the *steering strength*, $\sigma_T^\ell := \text{std}(h_T^\ell)$ is the local hidden-state scale (std. across the hidden dimension), and $s_\ell \in \{+1, -1\}$ orients the effect so that $+\alpha$ increases the permissive margin implied by P_{src} . We set s_ℓ with a one-shot sign probe: apply a small $+\alpha_{\text{probe}}$ (e.g., 0.8) on that layer and pair; if the measured margin shift Δ is opposite to the expected direction, flip the sign. When steering multiple layers simultaneously, we scale by $1/\sqrt{|\mathcal{L}|}$ to keep the intervention magnitude comparable.

Fallback direction. We compute a small, position-aware PCA “moral subspace” by applying PCA to last-token MLP activations across moral prompts and ranking components by how well their projections predict a permissive–restrictive margin. When a per-pair contrast is unavailable, we use the normalized average of the top-ranked PCA directions as a fallback and orient it with the same sign probe.

3.3 Calibration and Specificity

We estimate a *decision threshold* τ from neutral control prompts (e.g., weather/geography). We apply the same layers and a tempered subset of strengths ($|\alpha| \leq 1$) to controls and set τ to the empirical 95th percentile of $|\Delta|$ aggregated over those controls and strengths. A trial on a moral pair is counted as a success only if

$$\Delta \cdot \text{sign}_{\text{expected}} > \tau,$$

where $\text{sign}_{\text{expected}} \in \{+1, -1\}$ encodes the stance implied by the counterfactual source, computed by either a small NLI-based scorer or a lexical heuristic over the source completion.² All moral-pair interventions target the last token (`gate_on_mask=False`). For neutral controls we also gate the hook at the last token (`gate_on_mask=True`); specificity holds when steered control outputs exactly match their unsteered baselines. **Specificity metric:** we test that steering leaves unrelated content unchanged by regenerating short outputs for neutral controls with and without hooks and reporting *control preservation* (canonicalized exact match) and *corruption* rates.

3.4 Neuron-Level Steering

For each $\ell \in \mathcal{L}$, we select top- k neurons by absolute contrast magnitude at the decision position:

$$S_k^\ell = \text{top-}k(|h^\ell(P_{\text{src}})_{T,:} - h^\ell(P_{\text{base}})_{T,:}|).$$

We then add per-neuron nudges at the last token:

$$\tilde{h}_{T,j}^\ell = h_{T,j}^\ell + \alpha \cdot s_j^\ell \cdot \sigma_T^\ell, \quad j \in S_k^\ell,$$

with signs s_j^ℓ aligned by a single-neuron probe (flip if $+\alpha_{\text{probe}}$ yields a negative Δ). We evaluate $k \in \{1, 5, 10\}$ and test *reversibility* under $-\alpha$. The neuron-level success threshold is $\tau_{\text{neuron}} = \max(0.5\tau, 0.02)$ to reflect smaller expected effect sizes.

3.5 Implementation Notes

All interventions are implemented with standard hooks (no parameter updates): *forward hooks* for component-level steering and *forward pre-hooks* for neuron-level nudges. We target MLP modules of a decoder-only transformer, steer only the last token, normalize steering vectors with a device/dtype-safe ε (allocated on the module’s device and dtype), and scale by the local std. to reduce sensitivity to layer/dtype. We use greedy decoding for short baselines and read next-token logits directly to compute margins; the sweep selects the α with the highest calibrated success rate. An optional `split_variants` mode repeats the pipeline over disjoint prompt buckets for robustness.

4 Experimental Setup and Results

We now specify the concrete settings used to run the method in Section 3. This section details experimental setup and results.

4.1 Model and Hardware

We use **Meta-Llama-3-8B-Instruct**³ as the target model (decoder-only transformer). Unless otherwise noted, we run in `torch.float32` with greedy decoding (`do_sample=False`) and a fixed random seed (`seed=42`). Experiments were executed on two environments:

²*Implementation note.* The NLI scorer is only used to set the expected sign for orientation; the primary outcome is the next-token logit-margin shift.

³<https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct>

- **Local (Apple Silicon).** macOS device with Metal Performance Shaders (MPS) backend.
- **Colab (A100).** CUDA with bfloat16/float16 mixed precision for the base model; steering math (norms, scaling) uses the module’s compute dtype with device-allocated ε to remain dtype-safe.

We register forward hooks only; model weights are never updated.

4.2 Minimal-Pair Prompts

We evaluate on a set of *moral minimal pairs* that differ by a short phrase but imply opposite stances (permissive vs. restrictive). Examples include:

- “*Lying to save lives is morally ...*” vs. “*Lying to save face is morally ...*”
- “*Helping others is morally ...*” vs. “*Harming others is morally ...*”
- “*Stealing to save lives is ...*” vs. “*Stealing for personal gain is ...*”
- “*Breaking rules to help others is ...*” vs. “*Breaking rules to help yourself is ...*”

We instantiate $n=284$ minimal pairs by combining seed contrasts with a deterministic paraphrase (e.g., *morally*→*ethically*, insert “is generally”) and two light context wrappers (*In general, / From an ethical standpoint,*), then deduplicating and shuffling.⁴ For each pair $(P_{\text{base}}, P_{\text{src}})$, we generate a short greedy completion for P_{src} and use it to set the expected direction (Section 3.3) via a small NLI scorer or a lexical heuristic.

4.3 Control Prompts

To calibrate the decision threshold and measure specificity, we use neutral prompts unrelated to morality, e.g.:

- “*The weather today is ...*”
- “*Mathematics involves numbers and ...*”
- “*The capital of France is ...*”

For these prompts, we run the same hook configuration and compare steered vs. unsteered generations after canonicalization (lowercasing, symbol stripping). Control preservation is reported as exact-match rate.

4.4 Layers and Steering Vectors

We first run a lightweight diagnostic to identify influential components (position-aware PCA). For the component-level run we steer a fixed set of MLP layers spanning early and late blocks, while neuron-level tests focus on late layers. For the reported runs we steer *MLP* blocks at the following layers:

$$\mathcal{L} = \{0, 4, 8, 10, 16, 18, 20, 22, 28, 31\}$$

⁴If `data/minimal_pairs.csv` exists, we load it; otherwise we synthesize pairs as above. Expansion is deterministic and we set `seed=42` before sampling and shuffling.

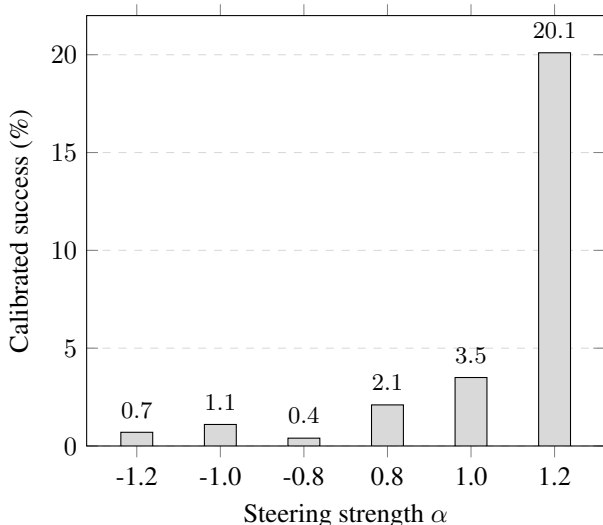


Figure 2: **Calibrated success across strengths.** Results on $n=284$ moral pairs, steering MLP layers $[0, 4, 8, 10, 16, 18, 20, 22, 28, 31]$; threshold $\tau=0.078$ from neutral controls. “Success” is the fraction of pairs with $\Delta \cdot \text{sign}_{\text{expected}} > \tau$. Counts per $\alpha \in \{-1.2, -1.0, -0.8, 0.8, 1.0, 1.2\}$: $[2, 3, 1, 6, 10, 57]$.

(0-indexed). For each pair and layer $\ell \in \mathcal{L}$ we compute a last-token contrast vector v_ℓ (Section 3.2); when unavailable, we fall back to a position-aware PCA direction computed from last-token MLP activations. Vector orientation is aligned per layer by a one-shot sign probe ($\alpha_{\text{probe}} \approx 0.8$) so that $+\alpha$ increases the permissive–restrictive margin, and we scale by $1/\sqrt{|\mathcal{L}|}$ when steering multiple layers. Neuron-level tests focus on late layers (e.g., $\{28, 31\}$). For moral pairs we always steer the last token; for neutral controls we use the same last-token position with mask gating so hooks are inert unless that position is active.

4.5 Strength Sweep and Decoding

We sweep a symmetric coarse grid

$$\alpha \in \{-1.2, -1.0, -0.8, 0.8, 1.0, 1.2\},$$

and scale the per-layer intervention by $1/\sqrt{|\mathcal{L}|}$ when steering multiple layers. Directions are oriented by a one-shot sign probe (see §3.2; $\alpha_{\text{probe}} \approx 0.8$) so that $+\alpha$ increases permissiveness. For each α and each pair, we run a single greedy step (`do_sample=False`) to read off next-token logits and compute the decision shift Δ . The calibrated success across strengths is plotted in Figure 2. We also log the sign of the margin change (permissive vs. restrictive) for summary counts.

4.6 Calibration Protocol

We estimate the decision threshold τ using the neutral control prompts from Section 4.3, following the procedure in Section 3.3. Calibration uses the same layers \mathcal{L} and a tempered subset of strengths ($|\alpha| \leq 1$). We compute $|\Delta|$ for every (control prompt, strength) combination and set τ to

the empirical 95th percentile of this pooled set; this τ is then fixed for the run and used to decide success on moral pairs. For neuron-level tests we use $\tau_{\text{neuron}} = \max(0.5\tau, 0.02)$.

4.7 Neuron-Level Configuration

For each $\ell \in \mathcal{L}$, we select top- k neurons by absolute contrast at the last token with $k \in \{1, 5, 10\}$. We apply per-neuron nudges using forward *pre*-hooks at the last token, aligning each neuron’s sign with a small probe. For $k=1$ we additionally test *reversibility* by applying $-\alpha$ at the same magnitude that yielded a positive flip.

4.8 Evaluation Metrics

We report the following:

- **Calibrated success rate:** percentage of pairs with $\Delta \cdot \text{sign}_{\text{expected}} > \tau$ at a given α . We select the best α on the sweep.
- **Direction counts:** number of permissive vs. restrictive shifts (sign of margin change) at the best α .
- **Control preservation / corruption:** exact-match rate on neutral controls (canonicalized) with/without hooks.
- **Neuron-level flips and reversals:** fraction of pairs that flip for $+\alpha$ (and flip back for $-\alpha$) at $k \in \{1, 5, 10\}$.

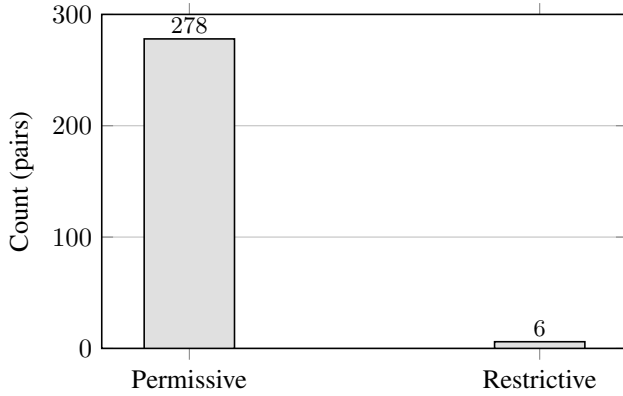
All metrics operate on next-token logits (no post-hoc reclassification of long completions).

4.9 Results

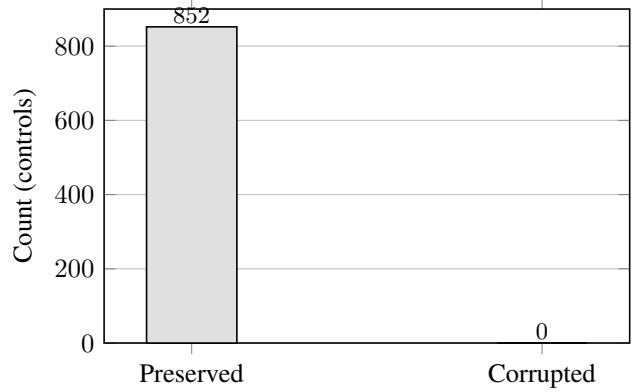
We evaluate on $n=284$ moral minimal pairs and a small pool of neutral controls, steering MLP layers $[0, 4, 8, 10, 16, 18, 20, 22, 28, 31]$ with a symmetric strength sweep. We report (i) a calibrated success rate, which counts a pair only if the next-token permissive–restrictive margin shift exceeds the control-derived threshold τ in the expected direction, and (ii) the directional effect (how often the shift is permissive vs. restrictive regardless of magnitude). We also measure specificity on controls and test neuron-level locality. General results for our targeted LLM (Llama-3-8B-Instruct) are summarized in Table 1.

Component-level steering. At the best strength on the sweep ($\alpha^* = 1.2$), the calibrated success rate is **20.1%** ($\approx 57/284$) under a control-derived threshold of $\tau = 0.078$. Directionally, **278/284** pairs shift permissive and only **6/284** restrictive, raising the permissive count from **253**→**278** (+8.8 percentage points). Specificity on neutral controls is **100%**: **852/852** exact matches and **0%** corruptions (three controls per moral pair; $3 \times 284 = 852$). Together these results show a reliable push in the intended direction with no detectable spillover to unrelated text. See Figure 3. With $k=5$ and $k=10$, we observe flips on *all* pairs in this run and clean reversibility: applying $+\alpha$ produces the shift and $-\alpha$ cancels it. For $k=1$, effects are smaller and more variable (flips **89.1%**, reversals **88.7%**). See Table 2. These results indicate that a compact set of late-MLP units is sufficient to control the decision margin while preserving specificity.

Interpretation. Thresholded success is conservative by design—many pairs move in the correct direction but remain below τ . Neuron-level edits concentrate causal mass and



(a) Directional effect at α^* : permissive vs. restrictive shifts.



(b) Specificity on neutral controls at α^* .

Figure 3: **Component-level results at best strength** $\alpha^* = 1.2$. MLP layers [0, 4, 8, 10, 16, 18, 20, 22, 28, 31]; threshold $\tau = 0.078$ from neutral controls. (a) Direction counts: 278/284 (97.9%) permissive vs. 6/284 (2.1%) restrictive. (b) Specificity: 852/852 exact matches (three controls per moral pair at α^* , i.e., $3 \times 284 = 852$).

avoid cross-layer cancellation, explaining their much higher flip and reversal rates compared to component-level mixing.

5 Limitations

Our claims are bounded by several design choices. This section summarizes the most important limitations of our experimental design.

Prompt scope and labels. We rely on *minimal pairs* that flip a permissive/restrictive stance by small textual edits. This operationalization is convenient for controlled experiments but does not cover the breadth of open-ended moral reasoning. Moreover, the polarity mapping (“permissive” vs. “restrictive”) is induced via a small set of lexical or NLI templates; alternative label spaces (e.g., deontic vs. consequentialist justifications) are not tested.

Last-token locality. All interventions are applied at the final input position. At the component level we steer a fixed set of MLP layers (including early/mid and late layers), while the neuron-level analysis focuses on late layers. This tests a specific hypothesis—that decisive evidence is consolidated at the decision token—but ignores earlier positions and cross-token computations. Effects that depend on multi-sentence context may be underrepresented.

Metric narrowness. We quantify shifts with a next-token *logit margin* between opposed continuations and use greedy decoding only for short baselines and control texts. This metric is simple and comparable across prompts but omits downstream decoding dynamics and multi-token rationales. The optional NLI scoring is template-based and model-dependent; it serves as a weak semantic check rather than a comprehensive evaluator.

Calibration sensitivity. The decision threshold is estimated from a small set of neutral controls. The resulting 95th-percentile cutoff can vary with the control pool, strength grid, and decoding policy (we use greedy decoding during calibration). Broader, category-balanced controls would yield a more stable baseline.

Selection/tuning bias. We select the best steering

strength α by sweeping on the same set of moral pairs we later summarize. This can inflate apparent success. A stricter protocol would separate a tuning split from a held-out evaluation split or use nested cross-validation.

Evaluation scale. Our evaluations use a moderate prompt set ($n=284$ moral minimal pairs) and a small pool of neutral controls. The pipeline emphasizes internal validity (directionality, reversibility, specificity), not exhaustive benchmarking. Broader claims would require larger and more diverse prompt pools, stronger statistics, and preregistered analysis plans. Specificity is measured as canonicalized exact-match of control generations (lowercasing and symbol stripping) under greedy decoding.

Model scope. We evaluate a single model (Meta-Llama-3-8B-Instruct), so all claims are specific to this setting. Generalization to other models and variants remains open. In follow-up work we plan to vary (i) *size* (smaller/larger Llama-3 variants), (ii) *instruction-tuned vs. base* checkpoints, and (iii) *model family/architecture* (e.g., Llama vs. Qwen/Mistral; dense vs. MoE) to test whether the observed late-layer locality, directional effects, and specificity persist.

6 Conclusions

We introduced a training-free, *contrastive last-token* steering method that modifies selected MLP outputs using directions derived from minimally different prompt pairs (evaluated here on Meta-Llama-3-8B-Instruct). At the component level (layers [0, 4, 8, 10, 16, 18, 20, 22, 28, 31]), the calibrated success rate at the best strength ($\alpha^*=1.2$) is 20.1% with $\tau=0.078$, while the directional effect is strong (278/284 permissive shifts) and control preservation is perfect (852/852). At the neuron level, steering top- k late-layer units yields near-universal flips and clean reversals under $\pm\alpha$ (100% for $k=5$ and $k=10$).

A calibrated logit-margin metric—with thresholds estimated from neutral controls—enables comparable measurement of directionality and effect size across prompts and strengths. In pilot neuron-level tests, small sign-aligned

Model	Layers (MLP, 0-index)	α^*	τ	Success	Base \rightarrow perm.	Steer \rightarrow perm.	Steer \rightarrow restr.	Spec.
Llama-3-8B Instruct	[0, 4, 8, 10, 16, 18, 20, 22, 28, 31]	1.2	0.078	20.1% ($\approx 57/284$)	253/284	278/284	6/284	100.0% (852/852)

Table 1: **Steering summary at best strength.** $n=284$ moral pairs; three neutral controls per pair. τ is the 95th percentile of $|\Delta|$ measured on controls. Specificity (Spec.) is exact-match control preservation (steered control outputs equal unsteered baselines).

Top- k	Flips @ $+\alpha$	Reversal @ $-\alpha$	n pairs
1	89.1%	88.7%	284
5	100.0%	100.0%	284
10	100.0%	100.0%	284

Table 2: **Neuron-level locality in late MLPs (last token).** Fraction (and counts) of moral pairs ($n=284$) that flip under $+\alpha$ and reverse under $-\alpha$ when steering only the top- k contrastive neurons in late layers $\{28, 31\}$. Evaluated at $\alpha^*=1.2$ with $\tau_{\text{neuron}} = \max(0.5\tau, 0.02)$ and $\tau=0.078$ from neutral controls.

subsets in late layers reproduced and reversed stance shifts, while neutral controls remained largely unchanged.

These observations indicate that, for our prompt set and model, parts of moral-stance formation can be *influenced* by small, interpretable perturbations at inference time. Immediate next steps include: (i) scaling evaluations with larger and more diverse prompt pools, (ii) generalization tests via split variants and out-of-distribution pairs, and (iii) ablations and placebos (random/shuffled directions, layer swaps, and component zeroing) to strengthen causal claims. Longer term, span-wise steering and cross-model replication can probe how localized these decision circuits are and how they interact with decoding policies and rationale generation.

References

- Abdulhai, M.; Serapio-Garcia, G.; Crepy, C.; Valter, D.; Canny, J.; and Jaques, N. 2023. Moral foundations of large language models. *arXiv preprint arXiv:2310.15337*.
- Elhage, N.; Hume, T.; Olsson, C.; Schiefer, N.; Henighan, T.; Kravec, S.; Hatfield-Dodds, Z.; Lasenby, R.; Drain, D.; Chen, C.; et al. 2022. Toy models of superposition. *arXiv preprint arXiv:2209.10652*.
- Hendrycks, D.; Burns, C.; Basart, S.; Critch, A.; Li, J.; Song, D.; and Steinhardt, J. 2020. Aligning ai with shared human values. *arXiv preprint arXiv:2008.02275*.
- Jiang, L.; Hwang, J. D.; Bhagavatula, C.; Bras, R. L.; Liang, J.; Dodge, J.; Sakaguchi, K.; Forbes, M.; Borchardt, J.; Gabriel, S.; et al. 2021. Can machines learn morality? the delphi experiment. *arXiv preprint arXiv:2110.07574*.
- Kim, B.; Wattenberg, M.; Gilmer, J.; Cai, C.; Wexler, J.; Viegas, F.; et al. 2018. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, 2668–2677. PMLR.
- Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegrefe, S.; Alon, U.; Dziri, N.; Prabhunoye, S.; Yang,

Y.; et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems* 36:46534–46594.

Meng, K.; Bau, D.; Andonian, A.; and Belinkov, Y. 2022. Locating and editing factual associations in gpt. *Advances in neural information processing systems* 35:17359–17372.

Nanda, N.; Chan, L.; Lieberum, T.; Smith, J.; and Steinhardt, J. 2023. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*.

Nangia, N.; Vania, C.; Bhalerao, R.; and Bowman, S. R. 2020. Crows-pairs: A challenge dataset for measuring social biases in masked language models. *arXiv preprint arXiv:2010.00133*.

Applying Relation Extraction and Graph Matching to Answering Multiple Choice Questions

Naoki Shimoda¹, Akihiro Yamamoto^{1,2}

¹Graduate School of Informatics, Kyoto University

²Center for Innovative Research and Education in Data Science,
Kyoto University

{shimoda.naoki.77s@st, akihiro@i}.kyoto-u.ac.jp

Abstract

In this research, we combine Transformer-based relation extraction with matching of knowledge graphs (KGs) and apply them to answering multiple-choice questions (MCQs) while maintaining the traceability of the output process. KGs are structured representations of factual knowledge consisting of entities and relations. Due to the high construction cost, they had been regarded as static databases with validated links. However, the recent development of Transformer-based relation extraction (RE) methods has enabled us to generate KGs dynamically by giving them natural language texts, and thereby opened the possibility for representing the meaning of the input sentences with the created KGs. Using this effect, we propose a method that answers MCQs in the “fill-in-the-blank” format, taking care of the point that RE methods generate KGs that represent false information if provided with factually incorrect texts. We measure the truthfulness of each question sentence by (i) converting the sentence into a relational graph using an RE method and (ii) verifying it against factually correct KGs under the closed-world assumption. The experimental results demonstrate that our method correctly answers up to around 70% of the questions, while providing traceability of the procedure. We also highlight that the question category has a vast influence on the accuracy.

1 Introduction

In this research, we combine **relation extraction (RE)** based on the Transformer and matching of **knowledge graphs (KGs)**, and apply the combination to answering multiple-choice questions where RE means to extract the relations of entities from given natural language texts. Since the construction of KGs required massive human annotations, KGs have long been regarded as static databases with only factually correct links. However, the recent development of Transformer-based RE methods has enabled us to construct KGs dynamically for any text in a natural language, e.g., English, at a low cost. KGs have a graph structure, and algorithms for graphs are applicable to them. In particular, deciding whether a KG matches another larger one as a subgraph semantically means the latter KG entails the former. Therefore, we conjecture that entailment of one sentence in natural language by another could be checked by applying RE to the sentences and graph matching to the obtained KGs. Moreover, the matching of KGs makes the entailment relation between the sentences traceable.

Multiple-choice questions (MCQs) are widely used in higher education due to their objective evaluation and the ability to conduct exams efficiently on a large scale. For learners who answered an MCQ test, receiving the reasons why each option is correct or incorrect after the test is effective for both enhancing the retention of the topic and reducing the misunderstanding of facts caused by reading the false options (Butler and Roediger 2008). However, it is not easy for teachers to create explanations for all the answers. Large language models (LLMs) are expected to reduce their burden by being asked to generate step-by-step explanations leading to the answers. Unfortunately, in the current status, it is known that such explanations often lead to reasoning errors depending on the properties of the options in MCQs and the order of them (Turpin et al. 2023). Moreover, even though becoming less often, LLMs might make hallucinations by stating non-existent facts, and reasoning based on them makes reliability a persistent issue.

Our study tries to overcome these problems by combining a Transformer-based RE method and matching of KGs, and proposes a method for answering MCQs with reliable explanations. The RE method creates a tailored relational graph for representing the meaning of the question texts of MCQs. By measuring the truthfulness of the obtained graph and verifying it against factually correct KGs under the closed-world assumption, our method outputs the most reliable word from the given choices with its explanation.

This paper is organized as follows. First, in Section 2, we provide the formal definition of the task that we addressed in this study. Section 3 introduces the key concepts and methods that underpin our discussion. In Section 4, we describe the details of the proposed method, followed by Section 5, where we present the results of answering experiments conducted on original MCQ datasets. Section 6 reviews related works in the fields of MCQ answering and fact verification. Finally, Section 7 concludes the paper and outlines directions for future work.

2 Multiple Choice Question Answering

Multiple-choice question answering (MCQA) is a task where a model is applied to select the correct option given a question sentence q , a set of options $\mathcal{O} = \{o_1, \dots, o_k\}$, and a set of background knowledge $\mathcal{C} = \{C_1, \dots, C_k\}$, where C_i is related to each choice o_i ($i = 1, \dots, k$) (Shah,

Gupta, and Roth 2020). We treat the case of the 4-way cloze test format, where the task is to choose the best word that fits in the blank from four options.

Example 1 (Barack Obama). *Let $q(x)$ be “ $\{x\}$ is an American politician and attorney who served as the 44th president of the United States from 2009 to 2017,” where*

$$\mathcal{O} = \left\{ \begin{array}{ll} o_1: \text{“Barack Obama”}, & o_2: \text{“Joe Biden”}, \\ o_3: \text{“George W. Bush”}, & o_4: \text{“Bill Clinton”} \end{array} \right\}.$$

The correct answer is o_1 : “Barack Obama”.

Definition 1 (Cloze test 4-way MCQA). *Cloze test 4-way MCQA is a task of selecting the correct option o_i given a question sentence $q(x)$ with a blank x , the set of four option words $\mathcal{O} = \{o_1, o_2, o_3, o_4\}$, and a set of background knowledge $\mathcal{C} = \{C_1, C_2, C_3, C_4\}$. The task is to find the correct answer $o_i \in \mathcal{O}$ such that*

$$C_i \models q(o_i), \quad (1)$$

where \models informally indicates the entailment between natural language sentences.

We require a transformation \mathcal{G} with which $q(x)$ and C_i are transformed into some objects $\mathcal{G}(q(x))$ and $\mathcal{G}(C_i)$, respectively, and to provide a **proof method** for deciding

$$\mathcal{G}(C_i) \vdash \mathcal{G}(q(o_i)),$$

to solve the entailment problem (1). We adopt the set of knowledge graphs (KGs) as the co-domain of \mathcal{G} because a KG is regarded as a set of conjunctions of binary ground atoms in first-order logic. Then the method of \vdash is the inclusion checking of two conjunctions, which can be seen as the subgraph isomorphism problem in graph terminology.

3 Preliminaries

In this section, we introduce the definition of KGs and the various concepts used in the proposed method.

A **Knowledge Graph (KG)** is a data structure that represents objects such as people, things, dates, and places as nodes and their relations as labeled directed edges (Kejriwal, Knoblock, and Szekely 2021). Focusing on the fact that a knowledge graph G consists of triplets (v_1, r, v_2) , where $r \in R$ represents the directed edge from node v_1 to v_2 , we represent G as a set of triplets $\{(v_1, r, v_2)\}$.

Relation Extraction (RE) is the task of extracting relations between entities described in a natural language sentence, and it is an important step to automatically construct KGs (Zhao et al. 2024).

Example 2 (Relation Extraction). *Assume that a sentence “Barack Obama, who served as the 44th President of the US, was born in Hawaii” is given, the RE is to output a set of semantic triplets, such as,*

$$\left\{ \begin{array}{l} (\text{“Barack Obama”}, \text{“born in”}, \text{“Hawaii”}), \\ (\text{“Barack Obama”}, \text{“is a”}, \text{“President of the US”}) \end{array} \right\}.$$

In this study, we consider two kinds of RE methods that are reported to achieve the best F1 scores in the NYT dataset (Riedel, Yao, and McCallum 2010; Zhao et al.

2024), namely **REBEL** (Huguet Cabot and Navigli 2021) and **UniRel** (Tang et al. 2022).

Entity Linking (EL) is the task of mapping a textual mention to its corresponding entry in a structured knowledge base (Shen, Wang, and Han 2015). Within the field of knowledge graphs, EL is widely used for **entity resolution**, i.e., merging nodes that represent the same real-world entity either within a single KG or across multiple KGs (Sun et al. 2018).

Semantic Textual Similarity (STS) is a measure of semantic equivalence between the given two blocks of text (Chandrasekaran and Mago 2021). In this study, we utilize the STS of different node labels as the **similarity of nodes** in KGs.

4 Methodology

Our proposing method takes a question sentence $q(x)$ and a set of option words \mathcal{O} as input and estimates the correct option $\hat{o} \in \mathcal{O}$ through the following steps:

1. Create a **propositional graph** $\text{PG}(o_i)$ for each choice o_i to represent the relations extracted from the sentence $q(o_i)$. We also create a **knowledge graph** $\text{KG}(o_i)$ for each choice o_i to represent the background knowledge for solving the question.
2. Estimate the corresponding nodes between $\text{PG}(o_i)$ and $\text{KG}(o_i)$.
3. Verify each triplet in $\text{PG}(o_i)$ against the ones from $\text{KG}(o_i)$ to compute the **truthfulness** of each original sentence $q(o_i)$ ($i = 1, 2, 3, 4$).
4. Choose the correct answer based on the truthfulness and the similarity of node labels between $\text{PG}(o_i)$ and $\text{KG}(o_i)$.

Through Steps 1 to 3, fact verification using KG matching is applied to each proposition $q(o_i)$, and the truthfulness of the proposition being true is computed by predefined rules. The final result is selected in Step 4. We assume that an RE method \mathcal{G} , an entity linking method \mathcal{L} , and a semantic similarity measure $\text{sim}(v_1, v_2)$ are pre-defined. In the following, we describe the details of each step.

Step 1.

Constructing Propositional Graph In the first step, we create a relational graph based on each proposition $q(o_i)$ to represent the relations to be verified. At this time, if o_i is an incorrect option, the relations represented in $q(o_i)$ will also contain factual errors. Therefore, we call the relational graph created from $q(o_i)$, which could contain incorrect relations, a **propositional graph** (PG), and distinguish it from the **knowledge graph** (KG) that only contains the ground truth triplets. By using an RE method \mathcal{G} , we construct such a propositional graph for each choice c_i ($i = 1, 2, 3, 4$) as $\mathcal{G}(q(o_i))$.

In the propositional graph for choice o_i , it is desirable that the structure of graphs other than the option label o_i be isomorphic for other choices. However, when using a machine learning-based RE method, the structure of the graph $\mathcal{G}(q(o_i))$ may differ for each sentence $q(o_i)$ because the

method has probabilistic behavior depending on the input. Therefore, to satisfy the aforementioned condition, we create a **propositional graph template** $\text{pg}(\#)$ by (i) replacing the node string o_i in original $\mathcal{G}(q(o_i))$ with a special symbol $\#$ and (ii) joining these $\mathcal{G}(q(\#))$ together:

$$\text{pg}(\#) := \bigcup_{o_i \in \mathcal{O}} \mathcal{G}(q(o_i)) \{o_i := \#\},$$

where $\{o_i := \#\}$ means the substitution of $\#$ to o_i . Hereafter, $\text{pg}(o_i)$, which is obtained by substituting o_i for the special symbol $\#$, is used as the propositional graph for each choice, instead of the original $\mathcal{G}(q(o_i))$.

Constructing Knowledge Graph Based on Wikipedia articles, we construct a knowledge graph for verifying the propositional graph. Here, let the set of nodes in $\text{pg}(o_i)$ be V_P . If a human were to verify the proposition $q(o_i)$, they would search for the keywords that are the subject of the sentence on Wikipedia and check if the content of the article matches the claim of the proposition. Our method mimics this process by treating each node label in $\text{pg}(o_i)$ as a topic. Therefore, for each node $v \in V_P$, its label $\text{label}(v)$ is searched in Wikipedia and the resulting article text $\text{text}(v | o_i)$ is converted into a relational graph.

$$\text{kg}(v | o_i) := \mathcal{G}(\text{text}(v | o_i))$$

$\text{kg}(v | o_i)$ for each node $v \in V_P$ is then joined together to construct the knowledge graph $\text{kg}(o_i)$ representing the ground truth triplets:

$$\text{kg}(o_i) := \bigcup_{v \in V_P} \text{kg}(v | o_i).$$

Resolving Node Label Ambiguity Because of the notational fluctuations in natural language texts, sometimes multiple nodes in the relational graphs represent the same entity in the real world. In order to resolve such node label ambiguity, we apply the entity linking method for each node in the constructed graphs and replace original labels with entity titles if found. By representing this process with the notation \mathcal{L} , the PG and KG for each choice o_i are given as

$$\text{PG}(o_i) = \mathcal{L}(\text{pg}(o_i)), \text{ and } \text{KG}(o_i) = \mathcal{L}(\text{kg}(o_i)).$$

Step 2. Let the node sets for $\text{PG}(o_i)$ and $\text{KG}(o_i)$ be V_P and V_K . In the following, we verify the correctness of the proposition $q(o_i)$ by comparing $\text{PG}(o_i)$ and $\text{KG}(o_i)$ created in Step 1. The verification is based on the idea that the more edges $(v_1, r, v_2) \in \text{PG}(o_i)$ are included in $\text{KG}(o_i)$, the more likely the original sentence $q(o_i)$ is correct.

However, since these graphs are created by RE, even nodes referring to the same entity may have variations in their labels at this time, depending on the original words in the sentences. Therefore, we need to find the corresponding nodes between the subset of V_P and V_K , namely $V'_P := V_P \setminus V'_K$ and $V'_K := V_K \setminus V_P$, respectively, excluding the nodes with perfectly matched labels. Then, we compute the mapping $\varphi': V'_P \rightarrow V'_K$ that maximizes the sum of sentence similarity between corresponding node labels:

$$\hat{\varphi}' := \arg \max_{\varphi'} \sum_{v \in V'_P} \text{sim}(\text{label}(v), \text{label}(\varphi'(v))).$$

Here, $\varphi': V'_P \rightarrow V'_K$ represents a **bijection** of node correspondences, and $\text{sim}(\cdot)$ represents the sentence similarity between two labels. This problem is a maximum matching problem in a bipartite graph consisting of two node sets V'_P and V'_K . Using the solution $\hat{\varphi}'$, the desired correspondence between the entire node sets $\hat{\varphi}: V_P \rightarrow V_K$ is defined as

$$\hat{\varphi}(v) := \begin{cases} \hat{\varphi}'(v) & \text{if } v \in V'_P, \\ v & \text{if } v \in V_P \cap V_K. \end{cases}$$

Step 3. Finally, we count the edges $(v_1, r, v_2) \in \text{PG}(o_i)$ that are also included in $\text{KG}(o_i)$, and define **truthfulness (edge score)** of the original proposition $q(o_i)$ as the ratio of mutually included edges.

We define a projected version of $\text{PG}(o_i)$, namely $\hat{\varphi}(\text{PG}(o_i))$, by replacing the nodes $v_1, v_2 \in V_P$ with the corresponding nodes $\hat{\varphi}(v_1), \hat{\varphi}(v_2) \in V_K$ as

$$\hat{\varphi}(\text{PG}(o_i)) = \{(v_1, r, v_2) \mid (v_1, r, v_2) \in \text{PG}(o_i)\}.$$

Using this notation, the desired truthfulness \mathcal{T} is expressed by the following equation:

$$\mathcal{T}(o_i) := \frac{|\hat{\varphi}(\text{PG}(o_i)) \cap \text{KG}(o_i)|}{|\text{PG}(o_i)|} \in [0, 1]$$

Here, $|G|$ represents the number of semantic triplets in a relational graph G .

Step 4. By applying the above three steps to $q(o_i)$ ($i = 1, 2, 3, 4$), we estimate the correct option

$$\hat{o} := \arg \max_{o_i \in \mathcal{O}} \mathcal{T}(o_i).$$

If there are multiple o_i that give the maximum value for $\mathcal{T}(o_i)$, we also take the average label similarity of the corresponding nodes $\mathcal{N}(o_i)$ into account in addition to $\mathcal{T}(o_i)$.

$$\mathcal{N}(o_i) := \frac{1}{|V_P|} \sum_{v \in V_P} \text{sim}(\text{label}(v), \text{label}(\hat{\varphi}(v)))$$

We call this $\mathcal{N}(o_i)$ the **node score**. If \hat{o} cannot be narrowed down to one, the method randomly selects an option from the choices that have the highest edge and node scores.

5 Experiments

In this section, we conduct answering experiments on original MCQ datasets to demonstrate the effectiveness and the traceability of our method. We also compare the performances of various RE methods over different lengths of question sentences to investigate the method's capability on each question category.

5.1 Experimental Setup

MCQ Datasets For the answering experiment, we created two variants of the MCQ dataset in the "fill-in-the-blank" format, namely KR-200m and KR-200s, using GPT-4o (OpenAI 2024). Both datasets consist of 10 categories shown in Figure 1, and each category has 20 questions. The difference between KR-200m and KR-200s is the length of the question sentence $q(x)$. KR-200m has 20.1 words in $q(x)$ on average, including blank position x counted as one word, while KR-200s has only 7.5 words per sentence.

	EL Setting	KR-200m	KR-200s
REBEL	w/ EL	53.5	48.0
	-	42.2	42.0
mREBEL ₄₀₀	w/ EL	52.2	49.7
	-	43.3	41.6
mREBEL ₃₂	w/ EL	49.0	43.6
	-	43.3	37.7
UniRel	w/ EL	28.6	26.9
	-	29.0	27.3

Table 1: Overall results of answering experiments by various RE methods, with and without entity linking (EL) methods applied in graph creation. The best results across different datasets and settings are highlighted in **bold**. Accuracy (%) is used as the evaluation metric.

Relation Extraction Methods For the used RE methods, we consider REBEL (Huguet Cabot and Navigli 2021), mREBEL₄₀₀, mREBEL₃₂ (Huguet Cabot et al. 2023), and UniRel (Tang et al. 2022) trained on the NYT dataset (Riedel, Yao, and McCallum 2010).

Entity Linking Method For the simplicity of the implementation, we construct an EL method using the title search of the Wikipedia API. In this method, we search for the original node labels in Wikipedia and link them to the article’s title, which the API returns as the most relevant. If no page is found, we do not assign any page title to the node, leaving the original node label.

Semantic Similarity Measure For the semantic similarity measure, we use a sentence embedding model all-MiniLM-L6-v2¹, which converts a sentence s into a 384-dimensional embedding vector, and define the similarity $\text{sim}(v_1, v_2)$ between nodes v_1 and v_2 as the cosine similarity of their embedding vectors. all-MiniLM-L6-v2 is a model trained on a dataset of over one billion sentences, based on the lightweight language model MiniLM (Wang et al. 2020).

Wikipedia Articles All Wikipedia pages are fetched on July 9th, 2025, via MediaWiki API endpoint². For creating KGs, we used the summary part of Wikipedia articles, which appears before any sections on the page.

5.2 Results

Overall Accuracy The overall results of answering experiments across various RE methods and EL settings are shown in Table 1. According to the table, REBEL and its variant mREBEL₄₀₀ achieve the highest accuracy on the two datasets, offering 53.5% and 49.7% accuracy on KR-200m and KR-200s, respectively. Following the two methods, mREBEL₃₂ performs at slightly lower accuracy, while UniRel merely answers with as much accuracy as a random

chance. These results show that each model’s number of relation types has an essential effect on accuracy. However, despite the vast difference between the number of relation types that REBEL and mREBEL₄₀₀ can generate (220 and 400, respectively), both models perform almost equally on the two datasets, indicating that too large a number of relation types does not improve the result anymore. In addition, the use of EL enhances the accuracy for three RE methods except UniRel, exhibiting the importance of node label disambiguation in the graph creation process.

Results by Category Figure 1 shows the accuracy per category for REBEL with EL enabled, which performed the best score on the KR-200m dataset. Among the 10 categories, our method achieved the highest accuracy of 71.7% on “Pop Culture”, followed by 7 categories with 50.0% to 66.7% accuracy. In contrast, the method did not perform well in “Mathematics” and “Science”. These results indicate that our method achieves higher accuracy in categories where factual knowledge, represented as semantic triplets, is required to select the best option. On the other hand, it does not perform well on questions that require abstract knowledge or logical inferences, such as mathematical derivations.

5.3 Case Study of Traceability

For a more intuitive understanding of the results, we illustrate PGs and KGs for an MCQ. The question is from KR-200m, “Art & Music” category, and the question sentence $q(x)$ and the set of option words \mathcal{O} are shown in Example 3.

Example 3 (Starry Night). *Let $q(x)$ be “Vincent van Gogh, a Dutch post-impressionist painter, created several masterpieces, including the famous painting called $\{x\}$,” where*

$$\mathcal{O} = \left\{ \begin{array}{l} o_1: \text{“Starry Night”}, \\ o_2: \text{“The Persistence of Memory”}, \\ o_3: \text{“Guernica”}, \quad o_4: \text{“The Scream”} \end{array} \right\}.$$

Figure 2 shows the PGs for the correct option o_1 and the incorrect option o_4 , respectively. Edges shown as solid lines are the ones verified from KGs. Comparing these two graphs, we can understand that the correct option o_1 : “Starry Night” was selected because all relations in $\text{PG}(o_1)$ are verified, whereas only one edge is verified for $\text{PG}(o_4)$. Through such a comparison between the PGs for the chosen option and the one not selected, our method offers traceability on why a certain choice was selected or **not** selected.

6 Related Works

In this section, we explain previous research on **multiple-choice question answering (MCQA)** and **fact verification**, which predicts if a given sentence is factually correct or not.

Most previous methods tackling MCQA naturally involve LLMs, because existing datasets, including MMLU (Hendrycks et al. 2021) and BIG-bench (BIG-bench authors 2023), are constructed for evaluating the performance of LLMs. On the other hand, a combination of KGs and language models has been proposed to enhance reasoning capabilities. A method is proposed by (Zhang et al. 2023) that

¹<https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>

²https://www.mediawiki.org/wiki/API:Main_page

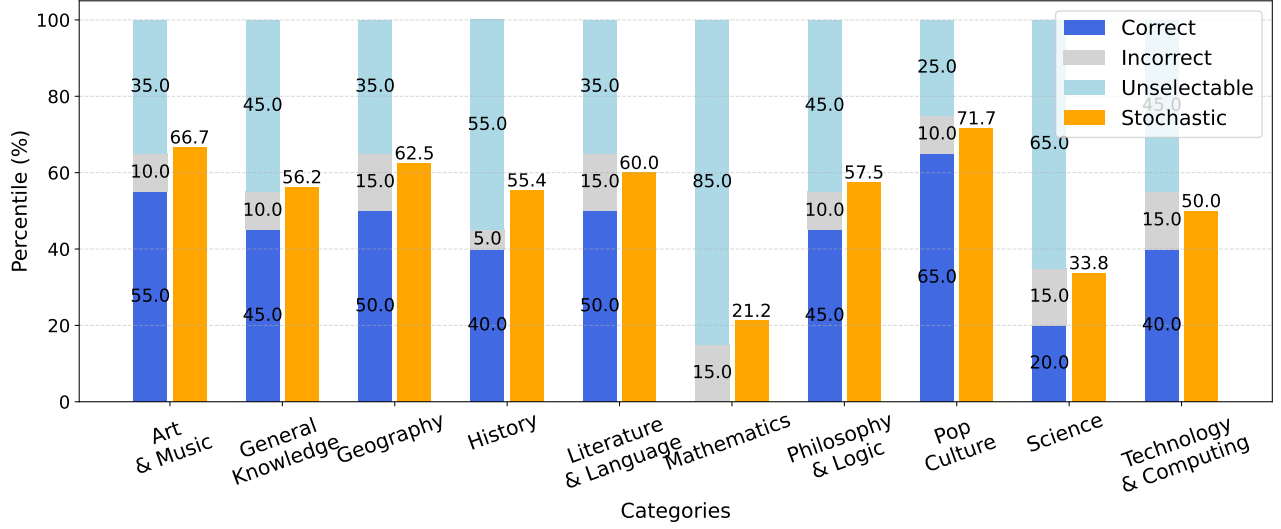


Figure 1: Accuracy per category for REBEL with entity linking on KR-200m. The columns with *Correct*, *Incorrect*, and *Unselectable* show the results without random choice, determined solely by the edge and node scores. *Stochastic* shows the accuracy with random selection for the unselectable questions.

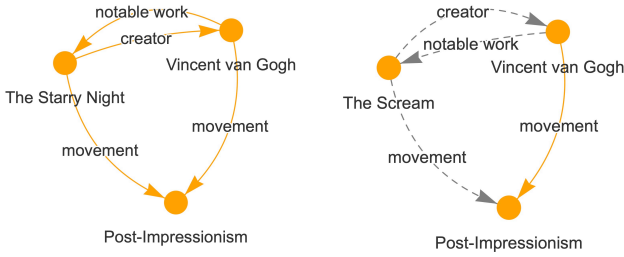


Figure 2: PGs for the **correct** option o_1 : “Starry Night” (left) and the **incorrect** option o_4 : “The Scream” (right).

retrieves subgraphs from a knowledge graph database tailored for the question sentence q and the set of options \mathcal{O} and inputs them into a Graph Neural Network (GNN) to obtain answers. This method is similar to our proposed method in that it creates KGs tailored to the problem setting. However, it differs from ours because it targets question formats other than the “fill-in-the-blank” and utilizes language models and reasoning mechanisms. Additionally, since it uses GNNs, there is a problem of low explainability regarding the process until the output results are obtained.

Fact verification is a task that determines whether a natural language sentence is factually correct, also known as fact-checking (Guo, Schlichtkrull, and Vlachos 2022). Although fact verification is a natural language task, language models lack clarity in the process leading to the answer, resulting in low explainability. Therefore, methods using KGs with explicit link structures have been proposed. Yuan and Vlachos (2024) proposed a method that extracts semantic triplets (subject, relation, object) from sentences of unknown veracity and verifies them against a knowledge graph database. This study aligns with ours in converting sen-

tences to be verified into a list of triplets. On the other hand, the method for preparing the ground truth knowledge graph differs, and they utilized additional reasoning modules.

Tackling MCQs on the cloze test format is positioned as a special case of MCQA. However, in MCQA, many previous studies use LLMs without the interest of interpretability, and methods pursuing traceability are in the minority. Therefore, in this study, we proposed an explainable MCQA method using KG-based fact verification.

7 Conclusion

In this paper, we presented a combination of the RE method and KG matching for answering MCQs in the “fill-in-the-blank” format. In our framework, the sentence, which is formed by substituting a choice word into the blank position of the question, is regarded as a proposition. We proposed a fact verification method by converting the sentence into a relational graph and verifying it against a factually correct KG. We conducted the answering experiments using various RE methods in two original MCQ datasets with different question lengths. Our method was able to answer up to 70% of the questions, depending on the category, while keeping the traceability of the chosen answer by visualizing verified edges in propositional graphs and the knowledge graph triplets used for verification. The result comparison over various RE methods reveals the tendency for a greater number of relation types in the RE method to cause higher accuracy. In addition, the ablation study demonstrates the importance of the entity linking method in the graph creation process. Future research could improve the accuracy of article search in the KG creation process in order to prevent the mislinking of Wikipedia pages and introduce a more flexible verification mechanism to implement reasoning capability in the method.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP25H01112. The authors would also like to thank Assistant Professor Nozomi Akashi for his valuable comments and feedback on the research.

Code and Data Availability

The source code and dataset used in this study are publicly available at: <https://github.com/N-Shimoda/KG-MCQA>.

References

- BIG-bench authors. 2023. Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.
- Butler, A. C., and Roediger, H. L. 2008. Feedback Enhances the Positive Effects and Reduces the Negative Effects of Multiple-choice Testing. *Memory & Cognition* 36(3):604–616.
- Chandrasekaran, D., and Mago, V. 2021. Evolution of Semantic Similarity—A Survey. *ACM Computing Surveys* 54(2):41:1–41:37.
- Guo, Z.; Schlichtkrull, M.; and Vlachos, A. 2022. A Survey on Automated Fact-Checking. *Transactions of the Association for Computational Linguistics* 10:178–206. Place: Cambridge, MA Publisher: MIT Press.
- Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Huguet Cabot, P.-L., and Navigli, R. 2021. REBEL: Relation Extraction By End-to-end Language Generation. In Moens, M.-F.; Huang, X.; Specia, L.; and Yih, S. W.-t., eds., *Findings of the Association for Computational Linguistics: EMNLP 2021*, 2370–2381. Punta Cana, Dominican Republic: Association for Computational Linguistics.
- Huguet Cabot, P.-L.; Tedeschi, S.; Ngonga Ngomo, A.-C.; and Navigli, R. 2023. RED^{fm}: a Filtered and Multilingual Relation Extraction Dataset. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 4326–4343. Toronto, Canada: Association for Computational Linguistics.
- Kejriwal, M.; Knoblock, C. A.; and Szekely, P. 2021. *Knowledge Graphs: Fundamentals, Techniques, and Applications*. Cambridge, Massachusetts: The MIT Press. OCLC: 1241215258.
- OpenAI. 2024. GPT-4o System Card. Version Number: 1.
- Riedel, S.; Yao, L.; and McCallum, A. 2010. Modeling Relations and Their Mentions without Labeled Text. In Hutchison, D.; Kanade, T.; Kittler, J.; Kleinberg, J. M.; Mattern, F.; Mitchell, J. C.; Naor, M.; Nierstrasz, O.; Pandu Rangan, C.; Steffen, B.; Sudan, M.; Terzopoulos, D.; Tygar, D.; Vardi, M. Y.; Weikum, G.; Balcázar, J. L.; Bonchi, F.; Gionis, A.; and Sebag, M., eds., *Machine Learning and Knowledge Discovery in Databases*, volume 6323. Berlin, Heidelberg: Springer Berlin Heidelberg. 148–163. Series Title: Lecture Notes in Computer Science.
- Shah, K.; Gupta, N.; and Roth, D. 2020. What Do We expect from Multiple-Choice QA Systems? In Cohn, T.; He, Y.; and Liu, Y., eds., *Findings of the Association for Computational Linguistics: EMNLP 2020*, 3547–3553. Online: Association for Computational Linguistics.
- Shen, W.; Wang, J.; and Han, J. 2015. Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. *IEEE Transactions on Knowledge and Data Engineering* 27(2):443–460.
- Sun, Z.; Hu, W.; Zhang, Q.; and Qu, Y. 2018. Bootstrapping Entity Alignment with Knowledge Graph Embedding. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, 4396–4402.
- Tang, W.; Xu, B.; Zhao, Y.; Mao, Z.; Liu, Y.; Liao, Y.; and Xie, H. 2022. UniRel: Unified Representation and Interaction for Joint Relational Triple Extraction. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 7087–7099. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics.
- Turpin, M.; Michael, J.; Perez, E.; and Bowman, S. 2023. Language Models Don’t Always Say What They Think: Unfaithful Explanations in Chain-of-Thought Prompting. In Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems*, volume 36, 74952–74965. Curran Associates, Inc.
- Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; and Zhou, M. 2020. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. In *Advances in Neural Information Processing Systems*, volume 33, 5776–5788. Curran Associates, Inc.
- Yuan, M., and Vlachos, A. 2024. Zero-Shot Fact-Checking with Semantic Triples and Knowledge Graphs. In Biswas, R.; Kaffee, L.-A.; Agarwal, O.; Minervini, P.; Singh, S.; and de Melo, G., eds., *Proceedings of the 1st Workshop on Knowledge Graphs and Large Language Models (KaLLM 2024)*, 105–115. Bangkok, Thailand: Association for Computational Linguistics.
- Zhang, Q.; Chen, S.; Fang, M.; and Chen, X. 2023. Joint Reasoning with Knowledge Subgraphs for Multiple Choice Question Answering. *Information Processing & Management* 60(3):103297.
- Zhao, X.; Deng, Y.; Yang, M.; Wang, L.; Zhang, R.; Cheng, H.; Lam, W.; Shen, Y.; and Xu, R. 2024. A Comprehensive Survey on Relation Extraction: Recent Advances and New Frontiers. *ACM Computing Surveys* 56(11):1–39.

FC-CONAN: An Exhaustively Paired Dataset for Robust Evaluation of Retrieval Systems

Juan Junqueras^{1*}, Florian Boudin², May Myo Zin³, Nguyen Ha Thanh^{3,4}, Wachara Fungwacharakorn³, Damián Ariel Furman¹, Akiko Aizawa⁴, Ken Satoh^{3,4}

¹Universidad de Buenos Aires, FCEyN, Departamento de Computación, Buenos Aires, Argentina

²JFLI, CNRS, Nantes University, Nantes, France

³Center for Juris-Informatics, ROIS-DS, Tokyo, Japan

⁴National Institute of Informatics (NII), Tokyo, Japan

Abstract

Hate speech (HS) is a critical issue in online discourse, and one promising strategy to counter it is through the use of counter-narratives (CNs). Datasets linking HS with CNs are essential for advancing counterspeech research. However, even flagship resources like CONAN (Chung et al., 2019) annotate only a sparse subset of all possible HS–CN pairs, limiting evaluation. We introduce **FC-CONAN** (Fully Connected CONAN), the first dataset created by exhaustively considering all combinations of 45 English HS messages and 129 CNs. A two-stage annotation process involving nine annotators and four validators produces four partitions—Diamond, Gold, Silver, and Bronze—that balance reliability and scale. None of the labeled pairs overlap with CONAN, uncovering hundreds of previously unlabelled positives. FC-CONAN enables more faithful evaluation of counterspeech retrieval systems and facilitates detailed error analysis. The dataset is publicly available¹².

Keywords: hate speech, counter-narrative, exhaustive annotation, fully paired dataset, dataset creation, benchmark creation, quality-graded partition, label sparsity, lower-bound bias, retrieval-system evaluation, recommender systems, information retrieval, bias in evaluation, evaluation metrics, annotation quality, argumentation, counterspeech, natural language processing (NLP).

1 Introduction

Disclaimer. This paper quotes hate speech verbatim for research purposes; some readers may find the language offensive.

Many Natural Language Processing (NLP) datasets consist of paired sentences, such as questions and answers (Rajpurkar et al., 2016), paraphrases (Dolan and Brockett, 2005), entailment (Bowman et al., 2015), translation (Koehn, 2005), and dialog (Li et al., 2017). While some datasets allow a single sentence to link with multiple others, such as CONAN (Chung et al., 2019), most cover only a fraction of all possible combinations. Exhaustive annotation is rarely attempted due to combinatorial growth and

cost, so unlabeled pairs remain ambiguous—often reflecting oversight rather than a true absence of relation. This incompleteness is especially problematic for recommendation tasks, where metrics can severely underestimate system performance (§4.3).

This challenge becomes particularly acute in domains such as hate speech. Social media has amplified the spread of harmful rhetoric (Silva et al., 2021), (Waseem and Hovy, 2016), prompting responses beyond content removal, which can reinforce censorship narratives. As a more constructive alternative, structured counterspeech protocols focus on timely, thoughtful responses that dismantle harmful arguments, avoid fostering further conversations, and align with broader goals. The CONAN dataset (COunter NAratives through Nichesourcing) (Chung et al., 2019), “the first large-scale, multilingual, expert-based dataset of hate speech/counter-narrative pairs”, remains the primary resource. To illustrate the nature of this data, we provide a representative example below.

Example 1 (An HS–CN pair from CONAN (Chung et al., 2019)). *HS (hate speech)* “I hate Muslims. They should not exist.”

CN (counter-narrative). “Muslims are human too. People can choose their own religion.”

Despite its widespread use, CONAN has notable limitations. **Crucially**, it does *not* annotate all HS–CN combinations, leaving many appropriate pairs unlabeled. This limits its usefulness, specifically for evaluating CN recommendation systems, as performance metrics reflect only a lower bound. In a pilot study using one of these systems, we found that while only 2 of 10 suggested CNs were labeled as appropriate, manual review judged 8 to be valid—highlighting the risk of underestimating system accuracy.

The lack of full HS–CN pair annotations also limits generation tasks by reducing training data for fine-tuning LLMs. Comprehensive HS–CN annotations would further enable methods such as contrastive learning. Ultimately, unannotated pairs leave valuable latent information unused, decreasing the dataset’s utility for downstream applications.

Another characteristic of the dataset is that the guidelines are rather open-ended. This approach stems from the fact that the original annotators had already been trained to

*Corresponding author: jjunqueras@dc.uba.ar

¹The dataset is publicly available at <https://github.com/jnqueras/FC-CONAN-dataset>

²This work was partially completed while the first author was at the National Institute of Informatics (NII), Tokyo, Japan.

follow NGO guidelines for crafting effective CNs. These guidelines are notably consistent across both languages and organizations, and closely mirror those established in the Get the Trolls Out project³. Annotators were encouraged to rely on their intuition, avoid overthinking, and produce reasonable responses (Chung et al., 2019, §3.2). It’s important to note that the high level of subjectivity is a characteristic of this field.

Due to resource constraints, we re-annotated a representative *subset* of all possible HS–CN pairs. This paper details that effort, originally motivated by the need to evaluate a counterspeech recommender more accurately. While CONAN covers three languages, our work focuses solely on English. Extending the annotation to other languages is future work.

2 Related work

In the hate speech domain, CONAN (Chung et al., 2019) is among the best-known multilingual resources. Expert-curated and focused on Islamophobia, it features hate speech (HS) and counter-narrative (CN) pairs in English, Italian, and French. Initially, included 4,078 pairs (1,288 in English) based on 136 unique HS messages, each matched with an average of 9.5 CNs. Through translation and paraphrasing, the English portion was expanded to 6,654 pairs, with 408 unique HS messages and 1,270 CNs. The dataset also includes metadata such as expert demographics, CN type, and HS sub-topic.

Several other datasets focus on hate speech and counter-narratives, such as DIALOCONAN (Bonaldi et al., 2022), which features multi-turn dialogues between a hater and an NGO operator, though it is not organized in HS/CN pairs. Another example is Multitarget CONAN (Fantón, Margherita and Bonaldi, Helena and Tekiroğlu, Serra Sinem and Guerini, Marco, 2021), a dataset with HS/CN pairs addressing multiple targets of hate. However, these datasets do not consider every possible combination of HS and CN pairs. From (Furman et al., 2023), one finding is most relevant to our work: a small LLM fine-tuned on a few hundred high-quality HS–CN pairs can outperform larger models. Our work complements theirs by focusing not on argumentative cues, but on exhaustively pairing HS–CN examples for robust evaluation and training.

A number of datasets have been developed for the task of hate speech detection, such as the Twitter corpus introduced by (Waseem and Hovy, 2016). Although our dataset could potentially be utilized for this purpose, it is primarily designed with a different focus.

3 Dataset Creation

We present FC-CONAN, a dataset of HS–CN pairs derived from a subset of the CONAN corpus. During annotation, we exhaustively considered all possible combinations of selected HS and CN items, labeling each pair as *appropriate* or *non-appropriate*, with some removed based on quality control criteria.

³<https://getthetrollsout.org/stoppinghate>

From the English partition of the CONAN dataset (the only language common to all annotators), we randomly selected 45 HS messages and collected the 375 CNs originally paired with them. For each CN, we used the SBERT model `all-MiniLM-L6-v2`⁴ to retrieve its two most similar CNs from the full CONAN dataset. We then discarded the 375 original CNs while retaining the HS messages and their newly retrieved CNs, ensuring that only novel HS–CN combinations were kept, so that annotators worked with CNs similar to the originals but not identical, preventing the task from being too easy. This process resulted in 133 unique CNs, from which we randomly selected 129 to ensure an even distribution of HS–CN pairs across annotators, producing 5,805 HS–CN pairs. Nine annotators—academically trained volunteers—labeled these pairs following adapted CONAN guidelines: fact-based information and maintaining a non-offensive tone to avoid escalating the conversation. Labels were assigned independently, with overlap enabling inter-annotator agreement checks, and adjudication resolved conflicts. Pairs marked *not sure* or irreconcilably disputed were discarded to avoid label bias, leaving 5,032 labeled pairs (4,143 as ‘*the CN is not appropriate for the given HS*’ and 889 as ‘*the CN is appropriate for the given HS*’).

To further ensure reliability, 4,000 adjudicated pairs underwent a validation round by four independent reviewers (not involved in initial annotation), each re-assessing 1,000 pairs. Validators applied the same guidelines, could skip distressing items (three pairs were skipped), and judged whether labels conformed. This process ensured every retained pair has both annotator and validator input. The validated pairs were retained, regardless of whether they were deemed valid or not. Pairs outside this set were discarded to prioritize label reliability over dataset size. Although some excluded pairs may still be appropriate, all possible combinations were reviewed during annotation.

The final resource balances reliability and coverage by defining four quality-graded subsets (*Diamond*, *Gold*, *Silver*, *Bronze*), allowing users to trade size for label confidence. Ethical safeguards included warnings, optional skipping, and on-demand debriefing breaks to support annotators. The dataset may pose dual-use risks if inverted to generate hateful replies; we therefore stress the need for responsible downstream use.

4 Results / Analysis.

We begin by describing the dataset itself before transitioning to system-level evaluation. Section 4.1 introduces the four quality-graded partitions generated through our annotation and validation pipeline. Section 4.2 then explores their internal structure. Finally, Section 4.3 presents a retrieval experiment that quantifies the impact of these partitions on downstream performance.

Partition	Total	Appr.	Non-Appr.
Diamond	551	35	516
Gold	663	54	609
Silver	3580	431	3149
Bronze	3997	702	3295

Table 1: Number of hate speech – counter-narrative (HS–CN) pairs in each dataset partition, categorized by appropriateness.

4.1 Dataset Partitions.

Following the annotation and validation processes (§3), we obtained HS–CN pairs annotated by one to three annotators and assessed for validity by one validator. Based on whether annotator labels aligned and the results of the validation process, we defined four distinct dataset partitions, each differing in annotation quality and size. Table 1 summarizes the size of each partition and the distribution of appropriate vs. non-appropriate HS–CN pairs. The characteristics of each partition are detailed below:

- **Diamond Standard Dataset:** This partition includes only HS–CN pairs annotated by two or more annotators who reached unanimous agreement—whether the counter-narrative was deemed appropriate or non-appropriate. Additionally, a validator has confirmed the accuracy of these annotations.
- **Gold Standard Dataset:** This partition extends the *Diamond Standard Dataset* by incorporating additional HS–CN pairs annotated by two or more annotators, regardless of whether the annotators unanimously agreed. In cases where annotators initially disagreed, these disagreements were resolved through the adjudication process (mentioned in § 3). Each resulting annotation was further reviewed and confirmed as accurate by a validator.
- **Silver Standard Dataset:** The *Silver Standard Dataset* includes all HS–CN pairs from the *Gold Standard Dataset*, along with pairs annotated by only one annotator and subsequently confirmed by the validators. Thus, annotations in this partition come from 1 to 3 annotators, with initial disagreements resolved via the aforementioned adjudication process. All annotations in this partition were approved during the validation stage.
- **Bronze Standard Dataset:** This dataset comprises all entries from the *Silver Standard Dataset*, supplemented by all the HS–CN pairs that were not approved during the validation stage. As before, annotations originate from 1 to 3 annotators, with disagreements resolved through the adjudication phase. However, unlike the Silver Standard, this partition also retains pairs that were not approved during the validation stage.

As expected, these partitions differ inversely in size and quality. Higher-quality datasets (Diamond and Gold) require greater annotation agreement and validation, resulting in smaller dataset sizes. Conversely, lower-quality datasets

(Silver and Bronze) are larger but contain annotations with potentially reduced reliability. Thus, the datasets, arranged from smallest to largest (highest to lowest quality), are: Diamond, Gold, Silver, and Bronze Standard datasets.

4.2 Qualitative Analysis of the Dataset

Across the six overlapping subsets described in Section 3, we obtain a mean Cohen’s $\kappa = 0.34$ ($\sigma \approx 0.20$), computed *before* any additional checks. As noted by Klie et al., “*although it is often treated as such, agreement is no panacea; high agreement does not automatically guarantee high-quality labels.*” (Klie, Eckart de Castilho, and Gurevych, 2024), so we applied the validation procedure outlined in Section 3. This section presents a qualitative analysis of the four dataset partitions: Diamond, Gold, Silver, and Bronze. Understanding these partitions helps users select the most suitable subset—prioritizing annotation reliability (Diamond/Gold) or volume (Silver/Bronze).

Table 1 confirms that many valid HS–CN pairs were unannotated in the original CONAN dataset. By evaluating new combinations, we recovered hundreds of appropriate and inappropriate pairs across all partitions.

Table 2 shows a trade-off between quality and quantity: Diamond and Gold are smaller but fully valid, Silver is larger and still fully validated, while Bronze is the largest yet includes some non-valid pairs—allowing users to choose between size and reliability for downstream tasks.

Table 3 illustrates the distribution of annotated pairs based on the number of annotators involved. Within the Bronze partition, of the 199 pairs annotated by three annotators, only 4 pairs ($\approx 2.01\%$) were deemed invalid by validators. For the 502 pairs annotated by two annotators, 34 pairs ($\approx 6.77\%$) were marked invalid. Lastly, among the 3,296 pairs annotated by a single annotator, 379 pairs ($\approx 11.49\%$) were classified as invalid. These observations indicate a clear trend: pairs annotated by multiple annotators tend to have proportionally fewer invalid instances, underscoring how reliability significantly improves with increased annotator agreement.

In sum, the four-tier partitioning balances label reliability. Diamond and Gold deliver perfect validation, making them ideal for benchmarking model performance under minimal label noise. Silver adds scale without compromising valid pairs, while Bronze boosts volume, introducing the only subset of non-valid pairs. Altogether, the new annotations help fill clear gaps in the original CONAN dataset.

4.3 Experimental Evaluation: Evaluation of Recommendation Systems.

As discussed in §1, incomplete CONAN labels masked appropriate CNs. In what follows, we re-evaluate recommendation systems on our exhaustively annotated FC–CONAN partitions⁵.

We compare twelve recommenders trained on an English-only dataset (*conan_not_in_bronze_train*) created by exclud-

⁴<https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>

⁵The datasets used in this experiment are available at https://github.com/jnqeras/FC-CONAN-dataset/tree/main/recommender_experiment_data

Partition	Valid	Non-Valid
Diamond	551	0
Gold	663	0
Silver	3580	0
Bronze	3580	417

Table 2: Number of hate speech – counter-narrative (HS–CN) pairs in each dataset partition, categorized by appropriateness.

Partition	3 annot.	2 annot.	1 annot.
Diamond	195	356	0
Gold	195	468	0
Silver	195	468	2917
Bronze	199	502	3296

Table 3: Number of HS–CN pairs annotated by 3, 2 and 1 annotators.

ing any HS or CN found in the Bronze partition, ensuring no overlap between training and evaluation. Specifically, we evaluate:

- **TF-IDF** – cosine similarity on TF-IDF vectors (sparse baseline); (Salton and Buckley, 1988).
- **BM25** – Okapi BM25 lexical ranker ($k_1=1.2$, $b=0.75$); (Robertson and Zaragoza, 2009).
- **random** – uniform sampling of ten candidate CNs.
- **sbert(MiniLM)** – Sentence-BERT *all-MiniLM-L6-v2* (384 d) + cosine; (Reimers and Gurevych, 2019), (Wang et al., 2020).
- **sbert(MPNET)** – Sentence-BERT *all-mpnet-base-v2* (768 d) + cosine; (Song et al., 2020).
- **sbert(RoBERTa-L)** – Sentence-BERT *all-roberta-large-v1* (1024 d) + cosine; (Zhuang et al., 2021).
- **textEmb3L** – OpenAI dense encoder *text-embedding-3-large* (3 072 d); (OpenAI, 2024b).
- **GPT-4o** – zero-shot prompting of *gpt-4o-2024-08-06* to return ten CNs; (OpenAI, 2024a).
- **RR25 system 1 / RR50 system 1** – top-25/50 CNs from textEmb3L, then GPT-4o re-ranks.
- **RR25 system 2 / RR50 system 2** – top-25/50 CNs from sbert(MPNET), then GPT-4o re-ranks.

To ensure a fair comparison, each system ranks a fixed candidate pool: the 100 unique CNs from the Bronze partition. This simulates a deployment where all models share the same, unchanging reply catalogue. To maintain consistency, we limit our query set to the hate speech instances in the Bronze partition (*bronze_HS*). For each HS in *bronze_HS*, each recommender produces its top-10 counter-narratives, yielding one ranking per model.

We then evaluate each ranking against all four partitions (Diamond, Gold, Silver, and Bronze). A recommendation is counted as correct if—and only if—the $\langle \text{HS}; \text{CN} \rangle$ pair is annotated as appropriate in the partition under test. As an example, we show the TF-IDF ranking for the following HS (capped at 2 CNs to keep the paper concise):

- “When Islam hasn’t arrived yet in England, things was good. Now we are full of muslim rapists and it is evident that sexual violence is jihad in England”.

The ranking is marked ✓ or ✗ for each partition to illustrate how appropriateness varies depending on which annotation standard we use.

1. Muslims aren’t rapists or murderers, in fact many Muslims are doing good things to fight the extremists falsely appropriating Islam: <https://5pillarsuk.com/2019/01/03/muslim-woman-praised-for-aiding-victims-of-new-years-eve-terror-attack-in-manchester/>. [D✗, G✗, S✗, B✗]
2. Rapists are, in most cases, friends, family, or partners of the victim. The proportion of Pakistani people who are rapists is no more than that of White British population. [D✗, G✓, S✓, B✓]

In the example ranking, the top-ranked counter-narrative (CN) is never judged appropriate in any partition. By contrast, the second-ranked CN is judged appropriate in the Gold, Silver, and Bronze partitions, but not in Diamond. This demonstrates that the evaluation metrics computed on a generated ranking can change substantially depending on which partition is used to define “appropriate” pairs.

Figures 1 and 2 show that recommender systems performance improves with partitions containing a greater number of annotated “appropriate” pairs: Diamond scores lowest, followed by Gold, then Silver, with Bronze achieving the highest values across both metrics. Overall, metric values scale roughly in proportion to the number of appropriate pairs in each partition. We observe the same trend for metrics such as NDCG@10, MAP@10, Precision@10, Accuracy@10, and F1@10, although their plots are omitted due to space limitations.

None of the pairs formed from the candidate pool (Bronze CNs) and *bronze_HS* are labeled as appropriate in the CONAN dataset; thus, using CONAN as the sole gold standard for this set yields zero scores across all metrics. As progressively more appropriate pairs are included—from CONAN’s subset to Diamond, Gold, Silver, and Bronze—metric scores increase. This suggests that when appropriate pairs remain unannotated, evaluation metrics serve only as lower bounds and fail to reflect true retrieval system performance.

Table 4 shows a clear performance hierarchy. **Embedding-based rankers** (all SBERT variants plus OpenAI’s textEmb3L) obtain the highest average score ($\mu \approx 0.32$) and the lowest coefficient of variation ($CV\% \approx 46$), indicating that dense vector representations are both *effective* and *robust* to missing annotations. **Hybrid rerankers** (RR50 system 1/2 and RR25 system 2) come next in terms of performance ($\mu \approx 0.23$) yet remain substantially less stable ($CV\% \approx 69$), presumably because the GPT-4o reranking step amplifies noise whenever the embedding pre-filter retrieves weak candidates. Among the **lexical baselines**, TF-IDF matches hybrid effectiveness ($\mu \approx 0.22$) while BM25 lags behind ($\mu \approx 0.18$); both exhibit high variability ($CV\% > 63$), confirming their sensitivity to annotation sparsity. The **LLM zero-shot** strategy (GPT-4o alone) clusters with the Hybrid reranker RR25 system 1

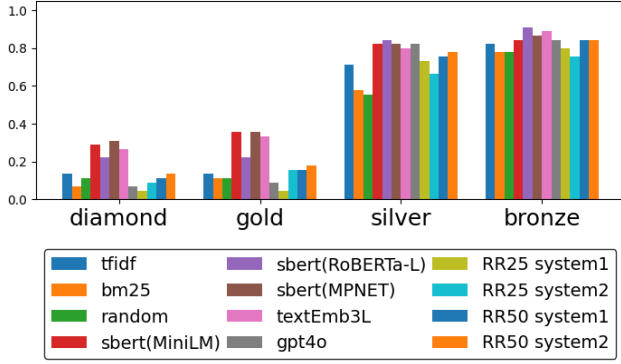


Figure 1: HIT RATIO@10 across the twelve systems (glossary in Section 4.3).

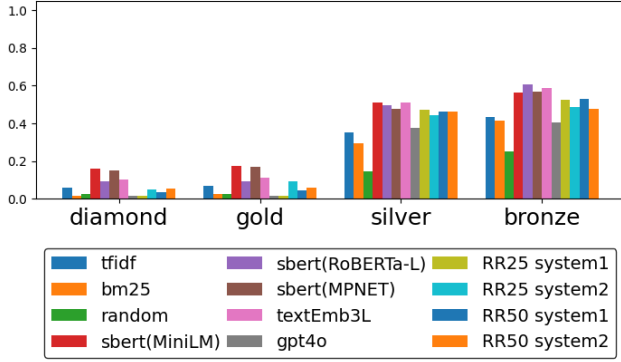


Figure 2: MRR@10 of the twelve evaluated systems (see glossary in Section 4.3).

($\mu \approx 0.21$, $CV\% \approx 89$). As expected, the **random** baseline sits at the bottom ($\mu = 0.15$). Overall, the results support the conclusion that *representation quality—rather than pipeline complexity alone—drives both effectiveness and robustness* in counter-narrative retrieval.

5 Limitations and Future Work

Generative fine-tuning and contrastive learning: Generative models can be fine-tuned on appropriate pairs from each partition to compare performance—we conducted such experiments but omitted them due to space limitations—while contrastive learning could leverage both appropriate and non-appropriate pairs to fully exploit the dataset’s structure.

Language scope: We cover only the English portion of CONAN. Extending exhaustive pairing to French and Italian remains future work.

Annotation coverage: Our 45 HS \times 129 CN subset produced partitions large enough to reveal evaluation artefacts, yet remains far from a *fully* exhaustive re-annotation of all possible combinations of CONAN. A semi-automatic “LLM-first, human-verify” pipeline could finish that job at lower cost.

Demographic diversity: Our annotator pool is skewed towards young, highly educated English-speaking individuals. Broader demographic sampling would reveal whether

System	Avg.	Min.	Max.	CV%
sbert(MPNET)	0.3283	0.1836	0.4904	38.425
sbert(MiniLM)	0.3270	0.1766	0.4804	39.200
textEmb3L	0.3128	0.1366	0.5098	53.200
sbert(RoBERTa-L)	0.3034	0.1214	0.5138	56.400
RR50 system2	0.2441	0.0658	0.4271	69.175
RR50 system1	0.2416	0.0526	0.4508	75.350
RR25 system2	0.2278	0.0514	0.3992	63.175
TF-IDF	0.2203	0.0703	0.4024	63.275
RR25 system1	0.2165	0.0184	0.4339	91.950
GPT-4o	0.2154	0.0267	0.4118	87.100
BM25	0.1819	0.0296	0.3810	81.650
random	0.1500	0.0414	0.3115	78.100

Table 4: Macro-level robustness of the 12 systems. For each system we average, over four metrics (HIT RATIO@10, MRR@10, NDCG@10, MAP@10), the metric-wise *mean*, *minimum*, *maximum* and coefficient of variation ($CV\% = (\text{standard deviation}/\text{mean}) \times 100$). Higher “Avg.” indicates better overall effectiveness, while lower CV% indicates greater stability across the Diamond–Bronze partitions.

cultural background influences appropriateness judgments.

6 Conclusions

We introduced **FC-CONAN**, to the best of our knowledge, the *first* hate speech / counter-narrative dataset where *every* possible pairing between two finite sets—45 HS messages and 129 CNs—is explicitly judged.

Consequently, if a pair is labeled *appropriate* in a given partition, it means that—according to the requirements of that partition—it was indeed deemed suitable. Conversely, if it is labeled *non-appropriate*, it reflects that it did not meet those same partition-specific criteria.

Four quality-controlled partitions—DIAMOND, GOLD, SILVER, and BRONZE—let practitioners trade annotation reliability for corpus size.

None of the HS–CN pairs we annotated occurs in the original CONAN corpus; our partitions therefore add hundreds of previously missing *appropriate* CNs, revealing how many unlabeled appropriate pairs the sparse labels in CONAN contained.

Importantly, using the partitions introduced, we confirm that not considering all possible pairs during dataset creation leaves many appropriate pairs unannotated. This results in artificially lower scores for most evaluation metrics in recommendation systems. For metrics that are not negatively affected by a greater number of positives—such as MRR@10 or Hit Ratio@10—this implies that scores obtained when evaluating on partially annotated datasets should be considered *lower bounds*.

We also showed that embedding-based rankers outperform others in both effectiveness and robustness to missing annotations.

In summary, researchers evaluating counter-narrative *retrieval* systems should rely on densely annotated datasets such as ours to avoid underestimating system performance.

Acknowledgements

This work was supported by the “R&D Hub Aimed at Ensuring Transparency and Reliability of Generative AI Models” project of the Ministry of Education, Culture, Sports, Science and Technology and JSPS KAKENHI Grant Numbers, JP22H00543. We also sincerely thank the annotators and validators who generously volunteered their time to contribute to this project. Junqueras was partially supported by the UBA BIICC Fellowship Program, the Fundar FunDatos Fellowship Program, and the NII International Internship Program.

References

- Abelson, H.; Sussman, G. J.; and Sussman, J. 1985. *Structure and Interpretation of Computer Programs*. Cambridge, Massachusetts: MIT Press.
- Basile, V.; Bosco, C.; Fersini, E.; Nozza, D.; Patti, V.; Rangel Pardo, F. M.; Rosso, P.; and Sanguinetti, M. 2019. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter. In May, J.; Shutova, E.; Herbelot, A.; Zhu, X.; Apidianaki, M.; and Mohammad, S. M., eds., *Proceedings of the 13th International Workshop on Semantic Evaluation*, 54–63. Minneapolis, Minnesota, USA: Association for Computational Linguistics.
- Baumgartner, R.; Gottlob, G.; and Flesca, S. 2001. Visual information extraction with Lixto. In *Proceedings of the 27th International Conference on Very Large Databases*, 119–128. Rome, Italy: Morgan Kaufmann.
- Bonaldi, H.; Dellantonio, S.; Tekiroglu, S. S.; and Guerini, M. 2022. Human-machine collaboration approaches to build a dialogue dataset for hate speech countering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 8031–8049. Association for Computational Linguistics.
- Bowman, S. R.; Angeli, G.; Potts, C.; and Manning, C. D. 2015. A large annotated corpus for learning natural language inference. In Márquez, L.; Callison-Burch, C.; and Su, J., eds., *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 632–642. Lisbon, Portugal: Association for Computational Linguistics.
- Brachman, R. J., and Schmolze, J. G. 1985. An overview of the KL-ONE knowledge representation system. *Cognitive Science* 9(2):171–216.
- Chung, Y.-L.; Kuzmenko, E.; Tekiroglu, S. S.; and Guerini, M. 2019. CONAN - COUNTER NARRATIVES THROUGH NICHE-SOURCING: A MULTILINGUAL DATASET OF RESPONSES TO FIGHT ONLINE HATE SPEECH. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2819–2829. Florence, Italy: Association for Computational Linguistics.
- Chung, Y.-L.; Abercrombie, G.; Enock, F.; Bright, J.; and Rieser, V. 2024. Understanding counterspeech for online harm mitigation. *Northern European Journal of Language Technology* 10:30–49.
- Chung, Y.-L.; Tekiroğlu, S. S.; and Guerini, M. 2021. Towards knowledge-grounded counter narrative generation for hate speech. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics.
- DataCanary. 2024. Quora question pairs.
- Davidson, T.; Warmley, D.; Macy, M.; and Weber, I. 2017. Automated hate speech detection and the problem of offensive language. *Proceedings of the International AAAI Conference on Web and Social Media* 11.
- Dolan, B., and Brockett, C. 2005. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*. Asia Federation of Natural Language Processing.
- Fanton, Margherita and Bonaldi, Helena and Tekiroğlu, Serra Sinem and Guerini, Marco. 2021. Human-in-the-Loop for Data Collection: a Multi-Target Counter Narrative Dataset to Fight Online Hate Speech. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.
- Furman, D.; Torres, P.; Rodríguez, J.; Martínez, L.; Alonso Alemany, L.; Letzen, D.; and Martínez, M. V. 2022. Parsimonious argument annotations for hate speech counter-narratives.
- Furman, D.; Torres, P.; Rodríguez, J.; Letzen, D.; Martínez, M.; and Alemany, L. 2023. High-quality argumentative information in low resources approaches improve counter-narrative generation. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2942–2956. Singapore: Association for Computational Linguistics.
- Gao, L., and Huang, R. 2017. Detecting online hate speech using context aware models. In Mitkov, R., and Angelova, G., eds., *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, 260–266. Varna, Bulgaria: INCOMA Ltd.
- Gillespie, T. 2018. *Custodians of the Internet: Platforms, Content Moderation, and the Hidden Decisions That Shape Social Media*. Yale University Press.
- Goffredo, P.; Basile, V.; Cepollaro, B.; and Patti, V. 2022. Counter-TWIT: An Italian corpus for online counterspeech in ecological contexts. In Narang, K.; Mostafazadeh Davani, A.; Mathias, L.; Vidgen, B.; and Talat, Z., eds., *Proceedings of the Sixth Workshop on Online Abuse and Harms (WOAH)*, 57–66. Seattle, Washington (Hybrid): Association for Computational Linguistics.
- Gottlob, G.; Leone, N.; and Scarcello, F. 2002. Hypertree decompositions and tractable queries. *Journal of Computer and System Sciences* 64(3):579–627.
- Gottlob, G. 1992. Complexity results for nonmonotonic logics. *Journal of Logic and Computation* 2(3):397–425.
- Klie, J.-C.; Eckart de Castilho, R.; and Gurevych, I. 2024. Analyzing dataset annotation quality management in the wild. *Computational Linguistics* 50(3):817–866.

- Koehn, P. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of Machine Translation Summit X: Papers*, 79–86.
- Levesque, H. J. 1984a. Foundations of a functional approach to knowledge representation. *Artificial Intelligence* 23(2):155–212.
- Levesque, H. J. 1984b. A logic of implicit and explicit belief. In *Proceedings of the Fourth National Conference on Artificial Intelligence*, 198–202. Austin, Texas: American Association for Artificial Intelligence.
- Li, Y.; Su, H.; Shen, X.; Li, W.; Cao, Z.; and Niu, S. 2017. DailyDialog: A manually labelled multi-turn dialogue dataset. In Kondrak, G., and Watanabe, T., eds., *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 986–995. Taipei, Taiwan: Asian Federation of Natural Language Processing.
- Nebel, B. 2000. On the compilability and expressive power of propositional planning formalisms. *Journal of Artificial Intelligence Research* 12:271–315.
- OpenAI. 2024a. Gpt-4o technical report. <https://openai.com/research/gpt-4o>.
- OpenAI. 2024b. New embedding models and api updates. <https://openai.com/index/new-embedding-models-and-api-updates/>. Accessed: 2025-08-02.
- Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In Su, J.; Duh, K.; and Carreras, X., eds., *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2383–2392. Austin, Texas: Association for Computational Linguistics.
- Reimers, N., and Gurevych, I. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 3982–3992.
- Robertson, S., and Zaragoza, H. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval* 3:333–389.
- Salles, I.; Vargas, F.; and Benevenuto, F. 2025. HateBRX-plain: A benchmark dataset with human-annotated rationales for explainable hate speech detection in Brazilian Portuguese. In Rambow, O.; Wanner, L.; Apidianaki, M.; Al-Khalifa, H.; Eugenio, B. D.; and Schockaert, S., eds., *Proceedings of the 31st International Conference on Computational Linguistics*, 6659–6669. Abu Dhabi, UAE: Association for Computational Linguistics.
- Salton, G., and Buckley, C. 1988. Term-weighting approaches in automatic text retrieval. *Information Processing & Management* 24(5):513–523.
- Silva, L.; Mondal, M.; Correa, D.; Benevenuto, F.; and Weber, I. 2021. Analyzing the targets of hate in online social media. *Proceedings of the International AAAI Conference on Web and Social Media* 10(1):687–690.
- Song, K.; Tan, X.; Qin, T.; Lu, J.; and Liu, T.-Y. 2020. Mpnnet: masked and permuted pre-training for language understanding. In *Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20*. Red Hook, NY, USA: Curran Associates Inc.
- Vallecillo Rodríguez, M. E.; Cantero Romero, M. V.; Cabrera De Castro, I.; Montejó Ráez, A.; and Martín Valdivia, M. T. 2024. CONAN-MT-SP: A Spanish corpus for counter-narrative using GPT models. In Calzolari, N.; Kan, M.-Y.; Hoste, V.; Lenci, A.; Sakti, S.; and Xue, N., eds., *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, 3677–3688. Torino, Italia: ELRA and ICCL.
- Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; and Zhou, M. 2020. Minilm: deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20*. Red Hook, NY, USA: Curran Associates Inc.
- Waseem, Z., and Hovy, D. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In Andreas, J.; Choi, E.; and Lazaridou, A., eds., *Proceedings of the NAACL Student Research Workshop*, 88–93. San Diego, California: Association for Computational Linguistics.
- Zampieri, M.; Malmasi, S.; Nakov, P.; Rosenthal, S.; Farra, N.; and Kumar, R. 2019. Predicting the type and target of offensive posts in social media. In Burstein, J.; Doran, C.; and Solorio, T., eds., *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 1415–1420. Minneapolis, Minnesota: Association for Computational Linguistics.
- Zhuang, L.; Wayne, L.; Ya, S.; and Jun, Z. 2021. A robustly optimized BERT pre-training approach with post-training. In Li, S.; Sun, M.; Liu, Y.; Wu, H.; Liu, K.; Che, W.; He, S.; and Rao, G., eds., *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, 1218–1227. Huhhot, China: Chinese Information Processing Society of China.

On the Role of Domain Experts in LLM-based Knowledge Formalization

Simon Vandeveld

Dept. of Computer Science, De Nayer Campus, KU Leuven, Belgium
Leuven.AI – KU Leuven Institute for AI
Flanders Make – DTAI-FET
s.vandeveld@kuleuven.be

Abstract

Modern Large Language Models (LLMs) offer many interesting new avenues for research in Knowledge Representation and Reasoning (KRR), due to their complementary natures. Most interestingly, they could offer a novel solution to the so-called Knowledge Acquisition Bottleneck, which arises from the difficulty in making accurate and reliable logical models of a problem domain. In literature, state-of-the-art research demonstrates how LLM-based formalization is already capable of translating small problem descriptions into formal notations. However, these works focus on a fully automatic process, without any input from external actors. In this position paper, we argue that domain experts should instead retain an active role in the formalization process to support knowledge validation. We introduce three complementary methods for doing so, and discuss their implementations and challenges.

1 Introduction

The recent surge in Large Language Model (LLM) development has opened up new possibilities within many scientific fields. In the field of knowledge representation and reasoning (KRR), researchers are actively looking at using LLMs to close the gap between state-of-the-art KRR and industry. Interestingly, both AI approaches are quite complementary: where LLMs are great at interpreting fuzzy text but struggle to faithfully reason over it¹, formal KRR engines can reliably reason over a problem domain, but only when given a precise, non-ambiguous encoding.

By fusing these approaches together, we can build systems that combine their advantages. For instance, LLMs can be used as a “front-end” for a reasoning engine, serving as a chat interface that can extract facts and perform queries (Rajasekharan et al. 2023; Callewaert and Vennekens 2024; Vakharia et al. 2024). They can also help improve explainability, by rewriting automatically generated explanations to be more understandable for laypeople (Vasileiou and Yeoh 2025). Lastly, LLMs could also help (partially) formalize problem knowledge by automatically translating natural language to a formal representation (Ishay, Yang, and Lee 2023).

¹whether or not they are capable of reasoning at all is still subject to much debate (Bender et al. 2021; Wu et al. 2024; Shojaei et al. 2025)

Crucially, this third method could offer a novel solution to one of the most significant challenges of KRR, namely the Knowledge Acquisition Bottleneck (Cullen and Bryman 1988). Knowledge Acquisition is the process of externalizing domain knowledge from experts and formalizing it in a formal model. As such, it is typically the first step in building knowledge-based systems. However, the process is known for being error-prone and time-consuming, hence the bottleneck. As shown in current research, LLMs can potentially automatically formalize knowledge, which would mitigate these issues and lift the bottleneck.

Though domain experts play a crucial role in traditional knowledge acquisition, current research on LLM-based formalization does not involve them. In this position paper, we argue that domain experts should instead play a central role in the process, and offer a few methods towards this idea. Put together, they form a novel methodology for human-AI knowledge formalization.

2 Knowledge Acquisition

When building knowledge-based systems, it is important to have an accurate and precise formal model of the problem domain at hand. For example, to build a legal reasoning tool, a knowledge engineer must correctly model a set of relevant laws so that a reasoning system can reason over them. However, this knowledge engineer is likely not well-versed in legal matters, and might find “legalese” (a.k.a. Legal English (Williams 2011)) hard to interpret. As a result, they decide to collaborate with a lawyer or notary to help them better understand the material, and to help them validate whether their tool works correctly.

This process is often referred to as *knowledge acquisition* (Leu and Abbass 2016), and is the first step in building knowledge-based tools. In brief, knowledge acquisition involves two parties: a domain expert with knowledge of a problem domain, and a knowledge engineer capable of modeling it. However, as both parties have diverging expertise, miscommunication will slip in quickly, making their collaboration time-consuming, error-prone, and labour-intensive. This difficulty is also known as the *Knowledge Acquisition Bottleneck* (Cullen and Bryman 1988), and is often cited as one of the main challenges for development of knowledge-based tools (Delgrande et al. 2024).

In an attempt to remove this bottleneck, research on

knowledge acquisition has spawned many new and unique methodologies (Leu and Abbass 2016). With the widespread availability of high-performant LLMs, the following question naturally arises: can LLMs auto-formalize knowledge for knowledge-based systems? Given that LLMs seem promising for writing imperative code such as Python or Java (Fan et al. 2023), generating models in a logical formalism does not seem far-fetched.

3 LLM-based Knowledge Formalization

In literature, we can already find multiple works in which LLMs generate logical models in one way or another. For instance, they might use an LLM to generate models in First Order Logic, Answer Set Programs (ASP) (Brewka, Eiter, and Truszczyński 2011), or Prolog (Bratko 2001), and use solvers such as clingo (Gebser et al. 2018), or IDP-Z3 (Carbonnelle et al. 2022). The existing literature on auto-formalization can be roughly divided into two groups based on their intention:

Formalize domain knowledge. The focus of the works in the first group is to generate a formal model based on a natural language description of a problem domain. These descriptions are typically quite general, without having an intended reasoning task in mind. Once a formal representation has been created, it can be fed by the user to a reasoning engine to solve problems in that domain. Examples of this group are (Goossens et al. 2023; Ishay, Yang, and Lee 2023; Mensfelt, Stathis, and Trencsenyi 2024; Coppolillo et al. 2024). An expected difficulty in generating these formal models is ensuring that they are syntactically correct. Here, (Mensfelt, Stathis, and Trencsenyi 2024) demonstrated how automatic checks can significantly improve on the syntactic correctness, while (Coppolillo et al. 2024) have shown that it is feasible to fine-tune the LLM for a specific formalism (in their case ASP), which also greatly improved the results.

Improve LLM reasoning. The aim of the works in the second group is to improve the logical reasoning of LLMs by giving them the capability of “outsourcing” their reasoning to a dedicated reasoning engine. When faced with a logical reasoning problem, they (a) convert it into a syntactically correct model, (b) execute the reasoning engine, and (c) convert the output back to natural language. Importantly, the conversion step (a) also takes into account the *type* of reasoning requested, which is a crucial difference with the first group. Indeed, some problem descriptions might require inductive reasoning, while others rely more on querying, optimization, or another reasoning type. Some examples of this approach are described in (Olafsson et al. 2023; Pan et al. 2023; Yang, Ishay, and Lee 2023; Callewaert, Vandevelde, and Vennekens 2025). When evaluated on reasoning benchmarks (e.g., (Tafjord, Dalvi, and Clark 2021; Zhong et al. 2021)), these neurosymbolic reasoners markedly outperform the baseline of LLMs with chain-of-thought reasoning.

While these prior works are already able to deliver promising results, they all share a major problem: they are unable to offer *formal guarantees* that their generated models are valid (i.e., that they actually correctly encoded the intended meaning of the problem description). This means that, when an LLM generates a seemingly correct solution, it might actually have major flaws – especially considering what we know about LLMs’ tendency to “hallucinate” incorrect knowledge. It is easy to imagine how a hallucinating LLM tasked with formalizing a piece of law text could have dire consequences if the resulting model is employed in a production environment unchecked.

Moreover, this issue is further exacerbated by domain size. In the aforementioned works, the problem descriptions in natural language were typically rather short, never exceeding more than a few paragraphs in length. Still, the systems in the second group were not able to exceed an accuracy of 90%. If an LLM cannot reliably generate valid models for small problem sizes, it will certainly struggle with larger ones where it will potentially compound the errors.

To make matters worse, manually finding such errors to fix the generated model is rather difficult. Interestingly, this difficulty shares a root cause with the knowledge acquisition bottleneck: domain experts cannot interpret the formal logics to ensure correctness, and knowledge engineers have insufficient knowledge of the problem domain to judge it. While one could devise a pipeline of test cases to automatically validate models instead, this requires such cases to be available and a test flow to be carefully designed.

For these reasons, we feel that a reliable, fully automatic translation from problem description to logical model is not yet possible with the current generation of LLMs, though it should be considered as a “holy grail”. Instead, we argue that, for realistic industry problems, reliable LLM-based formalization requires rigorous validation by keeping the domain expert in the loop.

4 Domain expert in the loop

As mentioned earlier, current state-of-the-art LLMs are not capable of formally guaranteeing the correctness of a model generated from natural language. Moreover, it is likely that they will never be able to do so altogether. Therefore, we believe the most compelling viable alternative to be validation by domain experts, where humans check that a logical model exhibits the correct behavior. This is not an easy task however, as we cannot expect domain experts to interpret the output of reasoning engines directly. Instead, validation for LLM-based formalization requires suitable tools and methodologies to support it. In what follows in this section, we outline three possible ways to involve the domain expert.

4.1 Visualisation and Interaction Tools

One promising method for validation by domain experts are tools that visualise solutions and offer interaction with the logical models. In this way, they can let domain experts “explore the formalized problem domain” and verify that the model’s behavior meets their expectations. For instance, after an LLM has generated a formal model of a (set of) legal

texts, a lawyer could try out some inputs and verify if the outputs correspond accordingly.

An example of such a tool is the Interactive Consultant (IC) (Carbonnelle et al. 2023), which is an interactive interface for knowledge bases in First Order Logic (FOL). It is fully generic in the sense that it automatically generates a GUI in which each variable of the logical model is represented by an input field. Users can assign values to these variables, after which the system derives the consequences. If a user is unsure why a consequence was derived, the system can also generate an explanation by pointing to the relevant user input and formulas in the knowledge base.

As an example, consider a small knowledge base on driver’s permits. In essence, this knowledge base² states that a person may only drive if they have their full permit or a learner’s permit (L2), which is only possible if they are over 18 or 16 years old respectively (L3-4).

```
1 theory {
2   drive() ⇔ permit() ∨ learners_permit().
3   permit() ⇒ age() ≥ 18.
4   learners_permit() ⇒ age() ≥ 16.
5 }
```

The image on the left in Figure 1 shows how the IC generates a tile for each symbol in the knowledge base, and allows the user to set specific values. In the image on the right, the IC shows the consequences of entering “age = 15”, i.e., that driving is not allowed. If the user asks why, it explains that driving requires one of two permits, neither of which are possible. In this way, the IC offers a very flexible interface for interacting with the formalized knowledge, allowing domain experts to validate it more efficiently. Though this is a small example, the IC has also proven itself useful for validation in larger past use cases (Aerts, Deryck, and Vennekens 2022; Vandeveld et al. 2024), with knowledge bases of up to 1600 lines of code.

Besides the IC, other existing tools that fit this idea are *clinguin* (Beiser, Hahn, and Schaub 2025), *clingraph* (Hahn et al. 2024), *ASP Chef* (Alviano and Rodriguez Reiners 2024) and the *Clafer* configurator (Bkak et al. 2014), though these require some configuration and are not fully automatic.

4.2 End-user formalisms

In its current form, LLM-based formalization has focused on going from a natural language description directly to the input language of a state-of-the-art solver, such as ASP or FOL. These formalisms are likely too technical for laypeople, who, as a result, cannot interpret them directly. Instead, we propose to have LLMs generate models in an “intermediary format” that is more user-friendly, which we refer to as *end-user formalisms* or *end-user notations*. In this way, the notation becomes more accessible for validation by domain experts.

For a formalism to be seen as an end-user formalism, it requires two important properties. Firstly, it should be designed with laypeople in mind, with the explicit focus of

being intuitive and interpretable by people without a logic background. Secondly, it should still be a precise formal language, which can be easily translated to the input language of an off-the-shelf reasoning tool.

A well-known class of such formalisms are Controlled Natural Languages (CNL), which restrict a natural language to a precise, non-ambiguous subset. Examples of such CNL are ACE (Fuchs, Kaljurand, and Kuhn 2008) and PENG (White and Schwitter 2009), which translate to first order logic, and CNL2ASP (Caruso et al. 2023), which translates to ASP. Though a small learning curve is inevitable, we can expect that knowledge formalized in CNLs will be easier to interpret (and thus, validate) by domain experts. To motivate this claim, the following listing shows a version of the driver’s permit knowledge base in a simple CNL:

```
1 theory {
2   drive() if and only if
3     permit() or learners_permit().
4
5   permit() requires age()
6     is greater than or equal to 18.
7   learners_permit() requires age()
8     is greater than or equal to 16.
9 }
```

As this formalism merely replaces the logical connectives with their natural language counterparts, there is a clear one-to-one translation from CNL into FOL. Yet, at the same time, these natural language connectives can improve interpretability for non-experts.

When targeting specific domains, we can also rely on domain-specific notations, which have the benefit of aligning closer to the natural intuition of a domain expert. For instance, Logical English (Kowalski 2020) is a CNL which has been specifically designed with the legal field in mind. Therefore, a piece of law text in Logical English will likely be more intuitive to legal professions than one in a more general CNL like ACE.

Note that end-user formalisms are not limited to CNLs, but that these simply lend themselves well to the text-based nature of LLMs. Other types of end-user formalisms tend to be more graphical, such as (c)DMN (Object Management Group 2021; Vandeveld, Aerts, and Vennekens 2021) and UML (Felfernig et al. 2002), which are likely harder to generate correctly.

4.3 Incremental Formalization

In current state-of-the-art LLM-based formalization, the translation from natural language to formal model happens in a single step. However, this is not possible for realistically large problem descriptions, such as the case for multiple law texts, due to three main reasons. Firstly, LLMs are limited in the number of input tokens they can handle per query, by way of a hard “context limit” that cannot be exceeded. Secondly, as explained earlier, LLMs struggle with formalizing larger problems, causing errors to slip in. Lastly, as the resulting model will be quite large, it will likely be challenging to validate all at once.

²Full example: <https://interactive-consultant.idp-z3.be/?file=permit.idp>

KU LEUVEN Interactive Consultant

age =

☒ ☒ drive

☒ ☒ permit

☒ ☒ learners permit

KU LEUVEN Interactive Consultant

age = 15

☒ drive

Above choice is implied by the following choice(s):

☒ age() = 15

Laws

- ☒ if permit(), then age() ≥ 18
- ☒ if learners_permit(), then age() ≥ 16
- ☒ drive() if and only if permit() or learners_permit()

Allow disabling laws (experts only!)

Figure 1: *Left*: IC without user choices. *Right*: IC explaining why drive is not possible when age = 15.

Instead of a single-shot approach, we argue to use an incremental formalization in which a formal model is built in multiple steps. This effectively decomposes the formalization in sub-problems, where the result of each sub-problem can straightforwardly be validated by a domain expert through the methods outlined earlier. For instance, a law text could be subdivided by its articles, resulting in bite-sized pieces of knowledge that can be more reliably converted into a formal model by an LLM and validated by the domain expert.

Additionally, incremental formalization also opens up possibilities for interactive chat sessions as a source of knowledge. This is useful in situations where knowledge is not (completely) written down. For instance, a design engineer often relies on many heuristics and rules-of-thumb which are not explicitly expressed anywhere. Through interactive chat sessions, they could still try to describe these tacit pieces of knowledge.

5 Challenges

We believe that joining all three ideas outlined in the previous section leads to a more robust and less error-prone formalization methodology. Nevertheless, introducing a domain expert in the loop brings new challenges, which we discuss here.

To begin with, involving people with non-IT backgrounds will inevitably increase the variability in result quality. For instance, we can expect that an engineer with a formal background in mathematics picks up the formalization process quicker than an administrative assistant without higher education. While both people will encounter a learning curve, it will undoubtedly be less steep for those with an IT, maths, or logics background. Therefore, it is important to select the proper tools for each situation, to ensure that they align well with the domain expert’s reasoning (e.g., Logical English for the legal domain).

Another challenge is the syntactic and semantic correctness of the LLM-generated models. Though state-of-the-art research shows steady improvements on this front, errors are

always bound to slip in. This is particularly true when using a “lesser known” language like the CNLs mentioned earlier, which appear less frequently (if at all) in the LLM training corpora. Possible solutions involve few-shot prompting, which showed promise in (Callewaert, Vandevelde, and Vennekens 2025), and further fine-tuning LLMs for the specific language, as demonstrated by (Coppolillo et al. 2024).

Depending on the problem domain, incremental formalization can also be tricky to get right. In “traditional” knowledge acquisition, experienced knowledge engineers typically develop an intuition on how to partition a large problem into smaller ones. Without such a knowledge engineer present, it will be up to the domain expert to do this correctly. As a possible solution, we could prompt LLMs to act as knowledge engineers and ask the right questions at the right moments. However, more research is needed to better estimate the feasibility of such an approach.

6 Conclusion

The recent advent of LLMs has opened up a number of interesting paths for KRR research. Through LLM-based formalization, these systems offer a promising approach to tackling the knowledge acquisition bottleneck, which has been a longstanding issue for knowledge-based applications. However, state-of-the-art research has so far exclusively focused on fully automatic formalization, without additional user input. Though these systems achieve encouraging results, they are limited to rather small problems which are not realistic for real-life use cases.

In this position paper, we argue for an active involvement of a domain expert in LLM-based formalization to validate the resulting logical models. We have outlined three possible methods towards this goal: (1) visualisation and interaction tools, (2) end-user formalisms, and (3) incremental formalization. Each of these methods focuses on putting domain experts in the driver’s seat when it comes to validation. Combined, they form a methodology which could enable domain experts to model their own domain knowledge with the help of LLMs.

Ethical Considerations LLMs are still surrounded by many ethical considerations. They exist in a grey zone w.r.t. foundational laws such as copyright, impose a high strain on energy and water consumption, and struggle with bias and transparency. These are important factors to take into account when designing LLM-based tools or workflows.

Acknowledgements The author thanks Nicholas Decleire for help in proofreading the paper, and the reviewers for their thoughtful and constructive feedback.

References

- Aerts, B.; Deryck, M.; and Vennekens, J. 2022. Knowledge-based decision support for machine component design: A case study. *Expert Systems with Applications* 187:115869.
- Alviano, M., and Rodriguez Reiniers, L. A. 2024. ASP Chef: Draw and Expand. In *Proceedings of the TwentyFirst International Conference on Principles of Knowledge Representation and Reasoning*, 720–730. International Joint Conferences on Artificial Intelligence Organization.
- Beiser, A.; Hahn, S.; and Schaub, T. 2025. Asp-driven user-interaction with clinguin. *Electronic Proceedings in Theoretical Computer Science* 416:215–228.
- Bender, E. M.; Gebru, T.; McMillan-Major, A.; and Shmitchell, S. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, 610–623. New York, NY, USA: Association for Computing Machinery.
- Bkak, K.; Diskin, Z.; Antkiewicz, M.; Czarnecki, K.; and Wasowski, A. 2014. Clafer: Unifying Class and Feature Modeling. *Software and Systems Modeling*.
- Bratko, I. 2001. *Prolog Programming for Artificial Intelligence*. Addison-Wesley, 3rd edition.
- Brewka, G.; Eiter, T.; and Truszczyński, M. 2011. Answer set programming at a glance. *Commun. ACM* 54(12):92–103.
- Callewaert, B., and Vennekens, J. 2024. ChatIDP: An Interactive Chatbot for IDP Knowledge Bases. In *Proceedings of the Benelux Conference on AI*.
- Callewaert, B.; Vandevelde, S.; and Vennekens, J. 2025. VERUS-LM: A versatile framework for combining llms with symbolic reasoning. In *Technical Communications of ICLP 2025*.
- Carbonnelle, P.; Vandevelde, S.; Vennekens, J.; and Denecker, M. 2022. IDP-Z3: A reasoning engine for FO(.).
- Carbonnelle, P.; Vandevelde, S.; Vennekens, J.; and Denecker, M. 2023. Interactive configurator with FO(.) and IDP-Z3.
- Caruso, S.; Dodaro, C.; Maratea, M.; Mochi, M.; and Riccio, F. 2023. CNL2ASP: Converting Controlled Natural Language Sentences into ASP. *Theory and Practice of Logic Programming* 1–31.
- Coppolillo, E.; Calimeri, F.; Manco, G.; Perri, S.; and Ricca, F. 2024. LLASP: Fine-tuning large language models for answer set programming. In *Proceedings of the TwentyFirst International Conference on Principles of Knowledge Representation and Reasoning*, KR-2024, 834–844. International Joint Conferences on Artificial Intelligence Organization.
- Cullen, J., and Bryman, A. 1988. The Knowledge Acquisition Bottleneck: Time for Reassessment? *Expert Systems* 5(3):216–225.
- Delgrande, J. P.; Glimm, B.; Meyer, T.; Truszczyński, M.; and Wolter, F. 2024. Current and future challenges in knowledge representation and reasoning (dagstuhl perspectives workshop 22282). *Dagstuhl Manifestos* 10(1):1–61.
- Fan, A.; Gokkaya, B.; Harman, M.; Lyubarskiy, M.; Sen-gupta, S.; Yoo, S.; and Zhang, J. M. 2023. Large language models for software engineering: Survey and open problems. In *2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FOSE)*, 31–53. IEEE.
- Felfernig, A.; Friedrich, G.; Jannach, D.; and Zanker, M. 2002. Configuration Knowledge Representation Using UML/OCL. In Jézéquel, J.-M.; Hussmann, H.; and Cook, S., eds., *UML 2002 — The Unified Modeling Language*, 49–62. Berlin, Heidelberg: Springer Berlin Heidelberg.
- Fuchs, N. E.; Kaljurand, K.; and Kuhn, T. 2008. Attempto Controlled English for Knowledge Representation. In Baroglio, C.; Bonatti, P. A.; Maluszyński, J.; Marchiori, M.; Polleres, A.; and Schaffert, S., eds., *Reasoning Web: 4th International Summer School 2008, Venice, Italy, September 7-11, 2008, Tutorial Lectures*. Berlin, Heidelberg: Springer Berlin Heidelberg. 104–124.
- Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2018. Multi-shot ASP solving with clingo. *Theory and Practice of Logic Programming* 19(1):27–82.
- Goossens, A.; Vandevelde, S.; Vanthienen, J.; and Vennekens, J. 2023. GPT-3 for Decision Logic Modeling. In *Proceedings of the 17th International Rule Challenge and 7th Doctoral Consortium @ RuleML+RR 2023 Co-Located with 19th Reasoning Web Summer School (RW 2023) and 15th DecisionCAMP 2023 as Part of Declarative AI 2023*. CEUR Workshop Proceedings.
- Hahn, S.; Sabuncu, O.; Schaub, T.; and Stolzmann, T. 2024. Clingraph: A System for ASP-based Visualization. *Theory and Practice of Logic Programming* 24(3):533–559.
- Ishay, A.; Yang, Z.; and Lee, J. 2023. Leveraging large language models to generate answer set programs. In *Proceedings of the 20th International Conference on Principles of Knowledge Representation and Reasoning*, 374–383.
- Kowalski, R. 2020. Logical english. In *Logic and Practice of Programming (LPOP) 2020*.
- Leu, G., and Abbass, H. 2016. A multi-disciplinary review of knowledge acquisition methods: From human to autonomous eliciting agents. *Knowledge-Based Systems* 105:1–22.
- Mensfelt, A.; Stathis, K.; and Trencsenyi, V. 2024. Autoformalization of game descriptions using large language models.

- Object Management Group. 2021. Decision model and notation v1.3.
- Olausson, T.; Gu, A.; Lipkin, B.; Zhang, C.; Solar-Lezama, A.; Tenenbaum, J.; and Levy, R. 2023. LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.
- Pan, L.; Albalak, A.; Wang, X.; and Wang, W. 2023. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. Association for Computational Linguistics.
- Rajasekharan, A.; Zeng, Y.; Padalkar, P.; and Gupta, G. 2023. Reliable Natural Language Understanding with Large Language Models and Answer Set Programming. *Electronic Proceedings in Theoretical Computer Science* 385:274–287.
- Shojaee, P.; Mirzadeh, I.; Alizadeh, K.; Horton, M.; Bengio, S.; and Farajtabar, M. 2025. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity.
- Tafjord, O.; Dalvi, B.; and Clark, P. 2021. ProofWriter: Generating Implications, Proofs, and Abductive Statements over Natural Language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 3621–3634. Association for Computational Linguistics.
- Vakharia, P.; Kufeldt, A.; Meyers, M.; Lane, I.; and Gilpin, L. H. 2024. ProSLM: A Prolog Synergized Language Model for explainable Domain Specific Knowledge Based Question Answering. In Besold, T. R.; d’Avila Garcez, A.; Jimenez-Ruiz, E.; Confalonieri, R.; Madhyastha, P.; and Wagner, B., eds., *Neural-Symbolic Learning and Reasoning*, 291–304. Cham: Springer Nature Switzerland.
- Vandeveld, S.; Aerts, B.; and Vennekens, J. 2021. Tackling the DM challenges with cDMN: A tight integration of DMN and constraint reasoning. *Theory and Practice of Logic Programming* 1–24.
- Vandeveld, S.; Vennekens, J.; Jordens, J.; Van Doninck, B.; and Witters, M. 2024. Knowledge-Based Support for Adhesive Selection: Will it Stick? *Theory and Practice of Logic Programming* 1–21.
- Vasileiou, S. L., and Yeoh, W. 2025. TRACE-CS: A synergistic approach to explainable course scheduling using llms and logic. *Proceedings of the AAAI Conference on Artificial Intelligence* 39(28):29706–29708.
- White, C., and Schwitter, R. 2009. An update on PENG light. In *Proceedings of the Australasian Language Technology Association Workshop 2009*, 80–88.
- Williams, C. 2011. Legal english and plain language: an update. *ESP Across Cultures* 8:139–151.
- Wu, Z.; Qiu, L.; Ross, A.; Akyürek, E.; Chen, B.; Wang, B.; Kim, N.; Andreas, J.; and Kim, Y. 2024. Reasoning or Reciting? Exploring the Capabilities and Limitations of Language Models Through Counterfactual Tasks. *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 1819–1862. Mexico City, Mexico: Association for Computational Linguistics.
- Yang, Z.; Ishay, A.; and Lee, J. 2023. Coupling Large Language Models with Logic Programming for Robust and General Reasoning from Text. *61st Annual Meeting of the Association for Computational Linguistics, ACL 2023* 5186–5219.
- Zhong, W.; Wang, S.; Tang, D.; Xu, Z.; Guo, D.; Wang, J.; Yin, J.; Zhou, M.; and Duan, N. 2021. AR-LSAT: Investigating analytical reasoning of text.

From Data Logs to Narrative Intelligence : Integrating ASP Logic and LLMs for Multi-Robot Missions

Shanmugaraj Madasamy^{1,5}, Karl Sammut^{1,5}, Richard Leibbrandt¹, Paulo Santos^{1,4}, Cedric Buche^{5,6}, Vincent Martin², Anne Gwenn Bosser³

¹Centre for Defence Engineering Research and Training, Flinders University, Australia

²Naval Group, France

³École Nationale d'Ingénieurs de Brest (ENIB), France

⁴Prior Analytica, Australia

⁵CNRS, IRL 2010 CROSSING, Australia

⁶IMT Atlantique, France

mada0090@flinders.edu.au, karl.sammut@flinders.edu.au, richard.leibbrandt@flinders.edu.au,
paulo.santos@priorianalytica.com, cedric.buche@cnrs.fr, vincent.martin@naval-group.com,
anne-gwenn.bosser@enib.fr

Abstract

Autonomous maritime vehicle missions produce large amounts of complex, low-level sensor data and irregular multi-agent communications, which can overwhelm mission operators and delay critical decision-making. Although large language models (LLMs) excel at natural language summarisation, they often lack the reasoning trustworthiness required for safety-critical environments. This study presents a framework that integrates Answer Set Programming (ASP) and LLM-based narrative generation to deliver logically sound and human-understandable summaries of multi-robot operations. Our prototype aims to process Autonomous Underwater Vehicle (AUV) sensor logs and agent communications, translate them into ASP code representations using GPT-4o, solve them using Clingo, and generate structured natural language explanations. We demonstrate how this pipeline supports both summary-only and LLM-ASP narrative modes. Initial experiments demonstrated improvements in situational awareness, traceability, and user comprehension. LLM-only summaries are insufficient for safety-critical missions. Trustworthy narratives need explicit, verifiable reasoning. Actions and events would be used to generate those meaningful narratives. We therefore pair PEC/ASP inference with LLMs, preserve provenance to data and rules, and assess correctness, temporal consistency, traceability, and operator workload.

1 Introduction

Maritime survey operations are increasingly incorporating teams of autonomous underwater vehicles (AUVs) to accelerate large-area seabed mapping, environmental monitoring, and search-and-rescue applications. AUVs can gather a diverse and large volume of sensor data, including Doppler Velocity Log (DVL), Inertial Measurement/Navigation Units (IMU/INS), depth, sonar, camera, and battery status information. Underwater communication is constrained to low-rate, intermittent acoustic links, while GPS and RF communications are only available when surfaced. Consequently, mission logs are noisy, asynchronous, and often contain missing or out-of-order entries.

1.1 Motivation

Mission operators are often overwhelmed by large volumes of heterogeneous data from distributed robotic agents, making it difficult to extract coherent explanations or traceable accounts of what occurred. While LLMs are highly capable of transforming raw data into readable narratives, they frequently lack the logical soundness and causal structure needed for dependable use in such contexts. Conversely, symbolic reasoning frameworks like ASP provide rigorous inferential capabilities but are not inherently designed to work with natural language or noisy, unstructured data. This motivates the development of hybrid neuro-symbolic systems that combine the strengths of both approaches, leveraging the generative flexibility of LLMs alongside the precision and auditability of symbolic logic to support transparent, robust understanding of complex, multi-agent operations.

1.2 Problem Statement

Autonomous maritime missions with teams of Autonomous Underwater Vehicles (AUVs) produce diverse sensor streams and inter-vehicle messages, often under limited bandwidth, variable latency, and intermittent links. Operators need timely, clear summaries in which each claim can be traced to the underlying data and rules. We define the task as generating such mission narratives from noisy, asynchronous logs.

We formalise *mission narrative synthesis* from multi-AUV logs, considering the problem over a time horizon T and data collected from n vehicles.

This data consists of:

- Sensor samples $S=(auv,t,k,v)$, e.g., DVL, IMU/INS, depth/pressure, sonar, battery
- Inter-vehicle/operator messages $C=(src,dst,t,msg)$. Clocks may differ by a bounded skew δt , δt , and data can be missing, delayed, or out of order.

A mission narrative $\mathcal{N} = \langle e_1, \dots, e_m \rangle$ of typed events with arguments and times, a set of temporal relations R between events (e.g., *before*, *during*), and a provenance map P linking each claim to supporting subsets of $S \cup C$ and to the rules used.

Variable sampling rates, acoustic-link latency/dropouts, and online operation under a latency budget L .

Find (\mathcal{N}, R, P) that maximises factual correctness, temporal consistency, and coverage of salient mission events, while guaranteeing traceability via P and meeting L .

1.3 Hypotheses

- H1 (Correctness): ASP-backed narratives achieve higher factual correctness and temporal consistency than LLM-only summaries on the same logs.
- H2 (Traceability): Operators recover supporting evidence for any narrative claim faster with provenance-linked ASP outputs than without provenance.
- H3 (Comprehension/Workload): ASP-backed narratives improve operator comprehension scores and reduce subjective workload relative to LLM-only.
- H4 (Robustness to noise): Under missing or out-of-order events, PEC+ASP maintains consistent conclusions more often than LLM-only summaries at matched noise levels.

2 Literature Review

Recent studies test LLMs on information and event extraction. They work reasonably well in open-ended settings, but their accuracy drops when a fixed schema, temporal links, or strict formatting are required Li et al. (2023); Gao et al. (2023a); Xu et al. (2024). These results suggest a practical split of responsibilities: use the LLM to help infer or align schemas from semi-structured inputs, but rely on a formal reasoner to validate and infer event relations.

LLMs can produce fluent narratives and summaries, yet faithfulness and temporal/causal consistency remain open problems Tian et al. (2024); Zhang, Liu, and Zhang (2023); Wu et al. (2023); Liang et al. (2024).

A growing line of work combines LLMs with tools or formal logic. ReAct interleaves reasoning with tool calls Yao et al. (2023), and PAL delegates exact computation to code generated by the model Gao et al. (2023b). Closer to our aims, recent papers couple LLMs with ASP: the model maps text to facts/rules, and the solver performs the logical inference, improving robustness and verifiability Yang, Ishay, and Lee (2023). Iterative frameworks (e.g., DSPy-style loops) further refine symbolic consistency Wang, Sun, and Kuhn (2024).

Recent studies explore causal, counterfactual, and contrastive explanations to support human situation awareness in mixed-initiative settings Gavrilidis et al. (2024). These styles help users answer “why,” “why not,” and “what if” questions. However, they are not generally tied to stable-model semantics or to provenance that links each claim back to concrete sensor/event evidence. We reuse these styles at the surface-language layer but ground their content in ASP/PEC inferences and explicit provenance.

Logic-based frameworks for event recognition provide relational and temporal structure Artikis et al. (2012). Probabilistic Event Calculus (PEC) encoded in ASP supports reasoning with inertia, initiation/termination, and uncertainty over complex event patterns Skarlatidis et al. (2015); Katzouris, Paliouras, and Artikis (2023). Maritime applications demonstrate the importance of temporal constraints and streaming data characteristics (asynchrony, delays) Ntoulias et al. (2022). These approaches, however, do not by themselves generate user-facing narratives; they require a language layer to render inferences clearly for operators.

Building on these lines, we combine

- LLMs for schema induction from semi-structured logs and for surface realisation of explanations
- PEC/ASP (and Telingo where needed) for verifiable temporal/causal inference.
- Provenance that maps each narrative statement to supporting data and rules.

This hybrid design aims to retain interpretability and auditability from symbolic reasoning while delivering readable narratives aligned with operator needs.

2.1 Event Recognition System

We adopt a logic-based event recognition approach in which joint actions are interpreted from a knowledge base encoded in Probabilistic Logic Programming (PLP) and grounded in a compact ontology of time, actions, fluents, and change Riguzzi (2023). We choose this because it provides explicit, auditable reasoning (rules and facts lead directly to conclusions), supports temporal and causal structure via Event Calculus/PEC with inertia and initiation/termination, and compiles to ASP for efficient inference under uncertainty Skarlatidis et al. (2015); Katzouris, Paliouras, and Artikis (2023). PLP also lets us attach probabilities to rules or facts, which helps handle noisy, delayed, and out-of-order AUV logs, and these rules can be learned from small labeled datasets and reused across missions Salam, Schwitter, and Orgun (2021).

2.2 Probabilistic Event Calculus

Event Calculus (EC) Mueller (2014) is a first-order logic-based approach for representing temporal information involving events, actions, and fluent dynamic properties that evolve over time. In this study Shanahan (2016), EC enables a compact representation of complex, high-level behaviours and offers a principled solution to the frame problem, by clarifying which aspects of the environment influence decision-making and which remain unaffected. The Probabilistic Event Calculus (PEC) has emerged as a powerful formalism for modeling and reasoning about events in autonomous maritime systems. Skarlatidis et al. (2015) proposed a probabilistic extension of the Event Calculus encoded into Answer Set Programming (ASP), enabling efficient relational probabilistic inferences in maritime environments. This approach facilitates the representation of the uncertainty and ambiguity inherent in maritime operations, thereby enhancing the event recognition capabilities of autonomous maritime vehicles.

2.3 Temporal Reasoning

Temporal reasoning is required to coordinate multi-vehicle missions and to reconstruct consistent timelines from asynchronous, delayed logs. This supports

- Synchronization and hand-offs between AUVs.
- Validation of rules under acoustic-link latency and dropouts,
- Elimination of explanations that violate timing constraints.

In maritime surveillance, Ntoulas et al. (2022) show that using temporal logic on streaming vessel data lets a system check prerequisites (e.g., task *A* must finish before task *B*) and model the impact of communication delays on mission progress.

2.4 Semantic Reasoning

When robots operate in the real world, it is not enough for them to recognize patterns or process raw sensor data; they also need to understand the context of those events. Semantic reasoning Liu (2022) is used to address this issue. It helps robots go beyond identifying what is happening, allowing them to grasp why it is happening and what it implies.

Liu et al. (2023) highlighted how adding semantic reasoning to robotic systems can make them more intelligent, reliable, and easier to understand. By tapping into structured knowledge, such as ontologies, semantic models, and knowledge graphs, robots can build a richer mental model of their environment. For example, if a robot sees someone picking up a toolbox, it does not just register the movement; it can infer that maintenance might be about to happen because it understands what a toolbox is used for and how it relates to human actions. Semantic reasoning also helps robots to deal with new or unexpected situations. If they encounter an unfamiliar object, they can reason by analogy, relating it to similar things they already know. This type of generalization is key in real-world settings, where things do not always go as planned.

2.5 Fusion of Multimodal Sensor Data

Fusion of multimodal sensor data: Integrating information from multiple sensors, including cameras, LiDAR, radar, GPS, and inertial sensors, enhances the richness and completeness of the data used for event recognition Tang, Liang, and Zhu (2023). Advanced fusion techniques, such as sensor fusion algorithms, Bayesian fusion methods, and information-theoretic approaches, enable robots to combine heterogeneous sensor data sources effectively, leading to more robust and comprehensive event recognition.

3 Proposition

3.1 Proposed Solution

We integrate LLMs with symbolic reasoning through a five-stage pipeline:

- Data normalization: parse AUV logs and communications into a time-stamped event table with units and schemas.

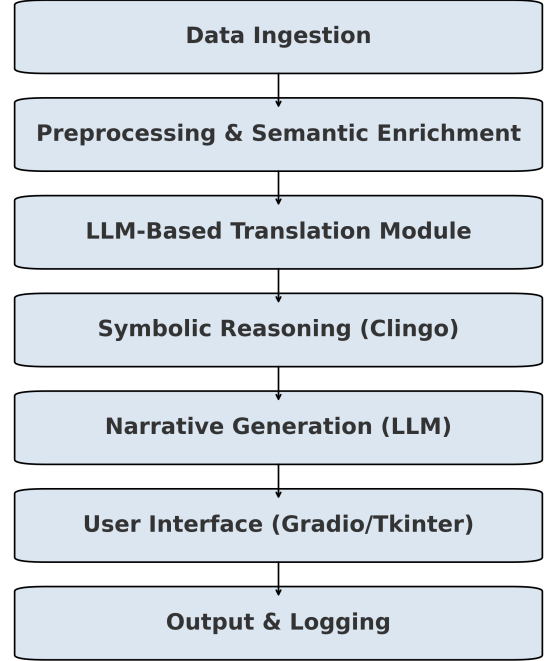


Figure 1: Narrative Pipeline

- ASP translation (LLM): use prompt-constrained GPT-4o to map events and domain statements into ASP predicates and rules (including PEC fluents and events).
- Symbolic inference (ASP/PEC): compute stable models with Clingo over domain axioms (inertia, initiation/termination, temporal constraints) and probabilistic annotations for uncertain observations.
- Narrative rendering (LLM): deterministically render the ASP-derived inferences into time-ordered, causally linked sentences; outputs include per-statement provenance to the generating rules and facts.
- Audit trail: persist inputs, ASP program, solver models, and narrative with rule/line references.

Safety-critical use requires verifiable inferences and reproducibility. ASP provides explicit rules, minimal stable-model semantics, and proof objects (grounded rule applications), enabling auditability. PEC encodes uncertainty while preserving explicit temporal and causal structure. The LLM is confined to two roles where it is effective and low risk: (i) schema induction from semi-structured logs to symbols, and (ii) surface realization from verified inferences. This separation reduces hallucination risk in conclusions and makes each step testable and replaceable.

3.2 Narratives

Narratives can be understood as a coherent ensemble of interrelated stories and are pervasive in activities that represent

ASP LLM Multi-Agent Interface

Agent/Robot Name:

Enter Conversation Log for this Agent:

Add Agent Log

Added Logs:

Agent: Alpha	Timestamp	Depth (m)	Temperature (°C)	Salinity (PSU)	Pressure (MPa)	Heading (°)	Speed (Knots)	Battery (%)	Turbidity (NTU)	Dissolved Oxygen (mg/L)	pH	Event
AUV_ID	9:00:00	247.5	8.7	35.21	25.07	123.1	2.18	100	1.62	5.36		
	7.95											Turbidity spike A18D-Alpha
	9:01:00	245.5	8.9	35.31	24.87	138.5	2.04	99.8	1.71	5.22		
	8.01											Sample collected A18D-Alpha
	9:02:00	247.4	8.5	35.21	25.06	166.6	2.1	99.6	1.66	5.28		
	7.99											Sample collected A18D-Alpha

Generation Mode:

☐ Generate with ASP ☒ Generate summary only

Process All Logs

Figure 2: Agent Event Logs

events over time Keith Norambuena and Mitra (2021). An event is something that happens or takes place, especially something significant or noteworthy, an incident, or an occurrence. Action is something done or performed, a deed, an act, habitual or ordinary deeds, and conduct. According to Ranade et al. (2022), narratives are stories told through speech, images, or writing, where events and characters are connected and usually unfold over time. Types of narratives: Visual, Textual and Domain-specific narratives.

The generated narratives may incorporate information such as location, heading, velocity, health status, detected objects, sensor readings (e.g., GPS, DVL, IMU, sonar), battery level, and water current characteristics (strength and direction). These narratives would be useful for human operators to assist in decision making during critical operations.

4 Experiment

We present an integrated GUI application designed to support both symbolic inference and natural language summarization for multi-agent robotic missions. The Input Layer takes manual or CSV-based ingestion of agent logs and AUV sensor data. Event grouping and preprocessing is done via a custom parser. In the LLM Translation Module, the GPT-4o used to translate conversations and sensor patterns into declarative ASP rules. This module prompts tailored to generate generalised, syntax-correct logic programs. At the ASP Processing Layer, Logic programs executed using Clingo to compute all stable models. Each answer set represents a possible consistent explanation of the mission. GPT-4o converts Clingo outputs into readable summaries i.e. Narratives. This module supports contrastive and causal explanation styles to enhance clarity. The output supports dual-mode, providing summary-only or ASP backed narratives. All components (inputs, ASP code, solver output, and narratives) saved as traceable .txt files for further reference.

We propose a system to simulated underwater rescue mission involving three AUVs. Sensor data and event logs were processed into logical assertions 4. The results illustrate the system’s ability to generate quick, consistent and meaning-

ASP LLM Multi-Agent Interface

Agent: Alpha

Timestamp Depth (m) Temperature (°C) Salinity (PSU) Pressure (MPa) Heading (°) Speed (Knots) Battery (%) Turbidity (NTU) Dissolved Oxygen (mg/L) pH Event

AUV_ID

9:00:00 247.5 8.7 35.21 25.07 123.1 2.18 100 1.62 5.36

7.95 Turbidity spike A18D-Alpha

9:01:00 245.5 8.9 35.31 24.87 138.5 2.04 99.8 1.71 5.22

8.01 Sample collected A18D-Alpha

9:02:00 247.4 8.5 35.21 25.06 166.6 2.1 99.6 1.66 5.28

7.99 Sample collected A18D-Alpha

Generation Mode:

☐ Generate with ASP ☒ Generate summary only

Process All Logs

LLM Input Prompt:

Please provide a clear, comprehensive summary of the following multi-agent conversation. Identify the key points, decisions made, and any outcomes or next steps discussed. Organize the summary in a structured way that highlights the main topics and conclusions.

LLM Output (Raw):

The conversation involves three autonomous underwater vehicles (AUVs) named Alpha, Bravo, and Charlie, conducting a mission with various data points recorded over a one-hour period. The key points, decisions, and outcomes are summarized below:

Extracted ASP Code:

Summary mode - No ASP code generated

Clingo Output:

Summary mode - No ASP output

Narratives:

The conversation involves three autonomous underwater vehicles (AUVs) named Alpha, Bravo, and Charlie, conducting a mission with various data points recorded over a one-hour period. The key points, decisions, and outcomes are summarized below:

Save All as .txt

Figure 3: ASP LLM Multi Agent Interface

ful narratives from raw mission data 3.

5 Evaluation Methodology

To validate the proposed framework, we conducted a controlled evaluation using mission logs generated from a simulated underwater rescue scenario (Figure 4). Three AUVs (Alpha, Bravo, Charlie) executed tasks including environmental sampling, navigation through restricted areas, and emergency response. The simulator produced multi-modal sensor data (depth, heading, speed, battery level, turbidity, dissolved oxygen) along with ground-truth mission events. Ground truth annotations were manually verified by domain experts. We compared two conditions:

- LLM-only baseline: mission logs were directly summarized by GPT-4o into natural language narratives without symbolic reasoning.
- Hybrid LLM+ASP: logs were first translated into ASP rules by GPT-4o, solved with Clingo/Probabilistic Event Calculus, and rendered into narratives with provenance links

The following metrics were computed:

- Event Detection Accuracy: the proportion of correctly identified mission-critical events relative to the ground-truth annotations.
- Temporal Consistency Error: the mean absolute difference (in seconds) between reconstructed timelines and ground truth.

Stage	Representation
Raw Log Entry	t=34, AUV Alpha, turbidity = 5.8 NTU (threshold exceeded)
ASP fact (LLM translation)	event(turbidity.spike, alpha, 34)
ASP rule (domain axiom)	initiates(collect.sample, A, T) :- event(turbidity.spike, A, T), threshold.exceeded(A, turbidity, T)
Solver inference (Clingo)	holdsAt(collect.sample(alpha), 34)
Narrative output (LLM)	“At 34 seconds, Alpha detected a turbidity spike and initiated a water sample collection.”

Table 1: Log Entry to Narrative

- **Narrative Coherence:** subjective ratings on a 1–5 Likert scale assessing completeness and causal clarity of generated narratives.
- **Operator Workload:** subjective workload estimates using a NASA-TLX–style instrument in a simulated evaluation setting.

Each log entry was traced through the pipeline to confirm provenance: from raw sensor record → ASP fact → solver inference → narrative output. 1 provides a worked example, linking an AUV turbidity event to its ASP encoding and final narrative. This methodology ensures that results are both quantitatively benchmarked against ground truth and qualitatively assessed for readability and operator usability.



Figure 4: Simulator Lechene et al. (2024)

6 Challenges

Despite significant advancements, several challenges remain in the development of event recognition systems for autonomous maritime vehicles. Signal processing of sensor data, real-time decision-making, and handling uncertainty in dynamic maritime environments continue to be areas of active research Arai et al. (2002). Several challenges remain in the development of event recognition systems for collaborative mission execution in autonomous maritime vehicles. These challenges include

```

### Summary of Multi-Agent Conversation

#### Participants:
- AUVs: A180-Alpha, A180-Bravo, A180-Charlie

#### Key Points:
1. **Turbidity Spikes:**
   - Multiple turbidity spikes were detected by all three AUVs at various timestamps, indicating potential disturbances or changes in water quality.
2. **Sample Collection:**
   - Samples were collected at several intervals by all AUVs, suggesting a focus on gathering data for analysis.
3. **Battery Checks:**
   - Regular battery checks were conducted, and all were reported as nominal, indicating no immediate concerns regarding power levels.
4. **INS Corrections:**
   - Inertial Navigation System (INS) corrections were applied at several points, ensuring accurate navigation and positioning of the AUVs.
5. **Sonar Anomalies:**
   - Sonar anomalies were detected multiple times, which may require further investigation to understand the cause.
6. **Grid Waypoint Reached:**
   - The AUVs reached several grid waypoints, indicating progress along a pre-defined survey path.
7. **Image Captures:**
   - Images were captured at specific intervals, likely for visual documentation or analysis.
8. **Holding Course:**
   - The AUVs maintained their course at various points, suggesting stable navigation.
9. **Mission Completion:**
   - All AUVs surfaced at 10:00:00, marking the completion of their respective missions.

#### Decisions Made:
- The AUVs continued their mission despite detecting turbidity spikes and sonar anomalies, indicating these were not considered critical enough to halt operations.
- INS corrections were applied as needed to maintain accurate navigation.

#### Outcomes:
- Successful data collection through samples and images.
- Completion of the mission with all AUVs surfacing as planned.

#### Next Steps:
- Analyze the collected samples and images to assess water quality and environmental conditions.
- Investigate the causes of sonar anomalies and turbidity spikes for potential environmental insights or operational adjustments.
- Review navigation data to ensure INS corrections were effective and consider any necessary improvements for future missions.

```

Figure 5: Narratives Generated using LLM

- Sensor fusion
- Real-time decision-making
- Robust communication
- Adaptive behavior in dynamic environments.

Additionally, integrating expert knowledge, ensuring system explainability, and addressing ethical considerations are critical for the deployment of these systems in real-world maritime scenarios.

Future research directions may focus on addressing these challenges, as well as exploring new methodologies and techniques to further enhance the capabilities and applications of event recognition systems for autonomous maritime vehicles. Collaborative research efforts involving interdisciplinary expertise from robotics, artificial intelligence, human-computer interaction, ethics, and social sciences will be essential for driving innovation and shaping the future of multi-robot systems Luo, Yih, and Su (2002).

7 Conclusion

This project propose to bring the symbolic reasoning and large language models together to help human operators to understand complex multi-robot missions. By combining the logic power of ASP with the LLM, we extend a new way to generate readable, meaningful narratives from complicated data. We believe this approach would contribute a step toward trustworthy AI systems that combine learning, reasoning, and communication.

Acknowledgments

This research project work was supported under work package 4: "Automatic report generation" proposed by the Naval Group Team at the 2022 IRL Crossing Workshop.

Abbreviation

AIS - Automatic Identification System
 APT - Advanced Package Tool
 ASP - Answer Set Programming
 AUV - Autonomous Underwater Vehicle
 CSV - Comma Separated Value
 DCA - Domain Closure Axioms
 DVL - Doppler Velocity Log
 GPS - Global Positioning System
 IMU - Inertial Measurement Unit
 KRR - Knowledge Representation and Reasoning
 LLM - Large Language Model
 OED - Oxford English Dictionary
 PEC - Probabilistic Event Calculus
 PLP - Probabilistic Logic Programming
 UNA - Unique Names Assumption

References

- Arai, T.; Pagello, E.; Parker, L. E.; et al. 2002. Advances in multi-robot systems. *IEEE Transactions on robotics and automation* 18(5):655–661.
- Artikis, A.; Skarlatidis, A.; Portet, F.; and Paliouras, G. 2012. Logic-based event recognition. *The Knowledge Engineering Review* 27(4):469–506.
- Gao, J.; Zhao, H.; Yu, C.; and Xu, R. 2023a. Exploring the feasibility of chatgpt for event extraction. *arXiv preprint arXiv:2303.03836*.
- Gao, L.; Madaan, A.; Zhou, S.; Alon, U.; Liu, P.; Yang, Y.; Callan, J.; and Neubig, G. 2023b. Pal: Program-aided language models. In *International Conference on Machine Learning*, 10764–10799. PMLR.
- Gavrilidis, K.; Konstas, I.; Hastie, H.; Munafo, A.; and Pang, W. 2024. Enhancing situation awareness through model-based explanation generation. In *The 2nd Workshop on Practical LLM-assisted Data-to-Text Generation*, 7–16. Association for Computational Linguistics.
- Katzouris, N.; Paliouras, G.; and Artikis, A. 2023. Online learning probabilistic event calculus theories in answer set programming. *Theory and Practice of Logic Programming* 23(2):362–386.
- Keith Norambuena, B. F., and Mitra, T. 2021. Narrative maps: An algorithmic approach to represent and extract information narratives. *Proceedings of the ACM on Human-Computer Interaction* 4(CSCW3):1–33.
- Lechene, H.; Clement, B.; Sammut, K.; Santos, P.; Cunningham, A.; Coppin, G.; and Buche, C. 2024. Lotus: Learning from operational teaming with unmanned systems. In *OCEANS 2024-Singapore*, 1–5. IEEE.
- Li, B.; Fang, G.; Yang, Y.; Wang, Q.; Ye, W.; Zhao, W.; and Zhang, S. 2023. Evaluating chatgpt’s information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness [arxiv: 2304.11633 [cs]]. *arXiv preprint arXiv:2304.11633*.
- Liang, X.; Wang, H.; Wang, Y.; Song, S.; Yang, J.; Niu, S.; Hu, J.; Liu, D.; Yao, S.; Xiong, F.; et al. 2024. Controllable text generation for large language models: A survey. *arXiv preprint arXiv:2408.12599*.
- Liu, W.; Daruna, A.; Patel, M.; Ramachandruni, K.; and Chernova, S. 2023. A survey of semantic reasoning frameworks for robotic systems. *Robotics and Autonomous Systems* 159:104294.
- Liu, W. 2022. A survey of semantic reasoning frameworks for robotic systems.(2022).
- Luo, R. C.; Yih, C.-C.; and Su, K. L. 2002. Multisensor fusion and integration: approaches, applications, and future research directions. *IEEE Sensors journal* 2(2):107–119.
- Mueller, E. T. 2014. *Commonsense reasoning: an event calculus based approach*. Morgan Kaufmann.
- Ntoulias, E.; Alevizos, E.; Artikis, A.; Akasiadis, C.; and Koumparos, A. 2022. Online fleet monitoring with scalable event recognition and forecasting. *GeoInformatica* 26(4):613–644.
- Ranade, P.; Dey, S.; Joshi, A.; and Finin, T. 2022. Computational understanding of narratives: A survey. *IEEE Access* 10:101575–101594.
- Riguzzi, F. 2023. *Foundations of Probabilistic Logic Programming: Languages, semantics, inference and learning*. River Publishers.
- Salam, A.; Schwitter, R.; and Orgun, M. A. 2021. Probabilistic rule learning systems: A survey. *ACM Computing Surveys (CSUR)* 54(4):1–16.
- Shanahan, M. 2016. The frame problem. *Stanford Encyclopedia of Philosophy*.
- Skarlatidis, A.; Paliouras, G.; Artikis, A.; and Vouros, G. A. 2015. Probabilistic event calculus for event recognition. *ACM Transactions on Computational Logic (TOCL)* 16(2):1–37.
- Tang, Q.; Liang, J.; and Zhu, F. 2023. A comparative review on multi-modal sensors fusion based on deep learning. *Signal Processing* 213:109165.
- Tian, Y.; Huang, T.; Liu, M.; Jiang, D.; Spangher, A.; Chen, M.; May, J.; and Peng, N. 2024. Are large language models capable of generating human-level narratives? *arXiv preprint arXiv:2407.13248*.
- Wang, R.; Sun, K.; and Kuhn, J. 2024. Dspy-based neural-symbolic pipeline to enhance spatial reasoning in llms. *arXiv preprint arXiv:2411.18564*.
- Wu, J.; Yang, S.; Zhan, R.; Yuan, Y.; Wong, D. F.; and Chao, L. S. 2023. A survey on llm-generated text detection: Necessity, methods, and future directions. *arxiv. arXiv preprint arXiv:2310.14724*.
- Xu, D.; Chen, W.; Peng, W.; Zhang, C.; Xu, T.; Zhao, X.; Wu, X.; Zheng, Y.; Wang, Y.; and Chen, E. 2024. Large language models for generative information extraction: A survey. *Frontiers of Computer Science* 18(6):186357.
- Yang, Z.; Ishay, A.; and Lee, J. 2023. Coupling large language models with logic programming for robust and general reasoning from text. *arXiv preprint arXiv:2307.07696*.

- Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.
- Zhang, H.; Liu, X.; and Zhang, J. 2023. Extractive summarization via chatgpt for faithful summary generation. *arXiv preprint arXiv:2304.04193*.