LLM-COGNICA: Towards Reliable Agent Decision Making

Adamos Koumi
Department of Computer Science
University of Cyprus
Cyprus
koumi.adamos@ucy.ac.cy

Antonis Kakas
Department of Computer Science
University of Cyprus
Cyprus
antonis@ucy.ac.cy

ABSTRACT

Following the recent development of LLMs it is suggested that these can now form the basis for Agentic AI. LLM-COGNICA is an approach to Agentic AI based on the integration of the Natural Language capabilities of LLMs with Argumentation-based Reasoning that is carried out on a Controlled Natural Language. It is a hybrid neural symbolic framework for developing decision making systems specified within Natural Language and executed via Explainable Argumentation. LLM-COGNICA offers the reliability of formal problem solving by allowing the human developer and problem solver to be in control of the agent system development process. The paper presents the LLM-COGNICA environment for the development of agent decision making, together with a systematic evaluation of the central element of mapping a decision policy in Natural Language to a corresponding policy in symbolic form.

KEYWORDS

Agentic AI, Large Language Models, Explainable Argumentation

ACM Reference Format:
Adamos Koumi and Antonis Kakas. 2026. LLM-COGNICA: Towards Reliable Agent Decision Making. In *AAMAS workshop NEXUS: Neurosymbolic explainable trustworthy Systems, Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 13 pages.

1 INTRODUCTION

Large Language Models (LLMs) are becoming ubiquitous in problem solving with the recent suggestion that they can now form the basis for Agentic AI. In such an approach, the agent's decision making will be driven by LLMs that are accustomed to an agent architecture, operating in Natural Language, both as the language of problem specification and user interface as well as the central language of computation. A hypothesis behind this suggestion is that LLMs possess or will soon possess a sufficiently good form of reasoning that would support the decision-making processes of agents.

In this paper, we consider an alternative complementary approach for enhancing the agentic problem-solving capability of LLMs. Instead of just trying to directly improve the reasoning capabilities of an LLM we can complement this effort by integrating together with LLMs symbolic systems that carry out (some of) the reasoning for the decision making needed in a problem-solving setting. In such a hybrid neuro symbolic framework we can then benefit both from the high quality of the natural language processing of LLMs together with the reliability and safeness properties of symbolic reasoning. In effect, we are proposing, like others [2, 6, 12, 25, 28], a hybrid LLM-REASONER approach as a programming or problem-solving framework from specifications and knowledge given in Natural Language text.

We present a concrete case of this hybrid approach to agentic AI, called LLM-COGNICA, based on the integration of LLMs with the formal symbolic framework of Computational Argumentation and in particular with the structured framework of Cognitive Argumentation (CA) [8]. Cognitive Argumentation has been developed from a synthesis between the study of human reasoning in Cognitive Science with the study of Argumentation in AI. The COGNICA [9] concrete realization of CA is one that concentrates on argumentation for conditional reasoning based on the mental models' approach proposed in [15] and extending this with further forms of modal reasoning together with hypothetical and default reasoning as exhibited in Human Reasoning. This synthesis is carried out via a connecting Controlled Natural Language (CNL) for Conditional Reasoning, based on the mental models theory proposed in [15]. The LLM-COGNICA framework sandwiches the COGNICA argumentation-based reasoning engine between an LLM, both for translating the problem requirements, e.g. an agent's Decision Policy, from their Natural Language form to the COGNICA Controlled Natural Language (CNL) for Conditional Reasoning, as well as to take the explainable reasoning of COGNICA that is used to compute the problem solutions and transform its explanation in a Natural Language form.The LLM-COGNICA framework web application is accessible at the following address: https://cognica.cs.ucy.ac.cy/COGNICAd/login.php.

The problem of training LLMs to translate problem specifications from Natural Language into executable form in some procedural programing language or symbolic reasoner is gaining a wide interest. Our approach to this problem is characterized by two main novel features (a) the target language of translation is still within Natural Language, albeit of a controlled form but still cognitively interpretable by humans, and (b) the executable symbolic reasoner is that of explainable argumentation. This "language to language" translation, together with the close connection of argumentation to human reasoning [22] facilitate the synthesis of LLMs with the symbolic reasoner of COGNICA and work together to give the approach an enhanced level of useability and reliability. They give the human developer an active control role in the problem-solving process. The potential non-reliability of using LLMs is isolated in the translation of the requirements which can be evaluated and corrected or clarified by the human developer. The cognitive CNL of COGNICA helps the developer distinguish between cases of unreliability of the "understanding" of the problem requirements by the LLM and cases where the statement of the requirements is unclear or ambiguous. Furthermore, normative requirements can be handled explicitly at the layer of the symbolic reasoner, in a

AAMAS workshop NEXUS: Neurosymbolic explainable trustworthy Systems, , May 25 – 29, 2026, Paphos, Cyprus.

reliable way based on the formal guarantees of the reasoner and its transparency that allows the solutions to be open to scrutiny by the developer.

The rest of the paper is organized as follows. In section 2 we present the Agentic AI problem setting and the main characteristics of the proposed LLM-COGNICA framework of integrating LLMs with symbolic reasoning. Section 3 describes the current form of the hybrid framework of LLM-COGNICA . It presents the target cognitive language of COGNICA , the process of customizing an LLM for translating a problem decision policy into COGNICA 's language and the LLM-COGNICA problem-solving environment. Section 4 presents a first systematic evaluation of the proposed LLM-COGNICA problem solving framework. Section 5 concludes with a discussion of the main directions of future work.

2 PROBLEM SETTING

We aim to develop a framework for intelligent, explainable and reliable agent problem-solving. The central element of an intelligent agent is that of a decision module governed by a policy that captures the requirements of operation of the module. Decision modules refer to various tasks of the agent, e.g., which goal to achieve and what plan to follow for this, or which other agents to collaborate with or protocols for carrying out a negotiation dialogue, etc. For some cases the decision policy provides guideline requirements and for other cases the policy defines more strict normative requirements that oversee the decisions of other task oriented policies.

In this paper, we will study the problem of reliably implementing this central element of decision making by bringing together, in a neural-symbolic framework, two AI technologies: that of LLMs and Argumentation-based Explanatory Reasoning. In this integrated framework we have the pipeline shown schematically in Figure 1.

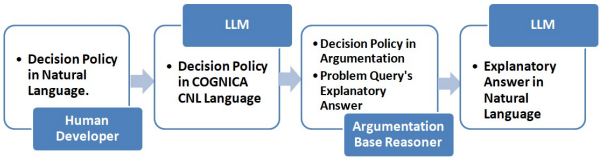

Figure 1: The LLM-ARGUMENTATION Pipeline

Starting from a Decision Policy expressed in Natural Language, this is translated using an LLM to a Control Natural Language (CNL) that is further translated to the formal language of the Argumentation REASONER. Problems are processed by the REASONER to give explainable decisions in a structured form of Natural Language, which are then paraphrased by the LLM into a more natural form.

Clearly, the choice of such a middle CNL is very important. It needs to be sufficiently rich to capture the policy requirements in their freer Natural Language form, but also allow formal logical reasoning to be carried out within it. Preferably, the form of reasoning within the CNL should also be close to that of Human Reasoning so that the implicit reasoning of the stakeholder drafting the decision policy can be more easily mapped into the CNL reasoning. We therefore require that the choice of CNL (a) has some form of Logical English [18] with a clear connection to logical reasoning and (b) is connected to Human Reasoning in Natural Language.

A candidate CNL is the language of human Conditional Reasoning as studied extensively in the area of Psychology of Reasoning within Cognitive Science. Following the work of [15] we will use a middle CNL that draws from their theory of conditionals and human reasoning with them. In our proposal, this mental models theory is captured via Computational Argumentation allowing the automation of conditional reasoning as well as the extension of their language of conditionals with other types of sentences.

Let us illustrate the CNL of COGNICA and the translation of policies into this via a simple example of a "requests handling" policy:

> " All requests should be made via the helpdesk. Any requests submitted through informal channels will be redirected to the helpdesk system except when the request is from a department manager . In general, requests made by department managers should always be accepted. Only requests made by collaborating agents will be accepted. Furthermore, requests without necessary approvals must be returned back."

In the current implemented version of the LLM-COGNICA framework the above policy would be translated via a trained LLM into:

(1) **IF** the request is via the helpdesk **THEN** accept request.
(2) **IF** the request is submitted through informal channels **THEN** redirect to the helpdesk system **UNLESS** the request is from a department manager **IN SUCH CASES** accept request.
(3) **IF** the request is from a department manager **THEN ALWAYS** accept request.
(4) **ONLY IF** the agent is a collaborator **THEN** accept request.
(5) **IF** NOT necessary approvals **THEN ALWAYS** return request.

This CNL form of the policy would be further automatically translated into a symbolic form, which is readable by the argumentation reasoner. Then given a problem scenario, e.g., a request via the helpdesk from a non-collaborator, the reasoner can reason with the policy to arrive at a decision and subsequent action, e.g., in the above scenario: "Reject. Request not from a collaborator agent".

2.1 LLM-COGNICA Argumentation Reasoner

As mentioned above, the symbolic reasoning framework of COGNICA is that of Computational Argumentation [3]. Computational Argumentation is a richly studied field in AI with many different proposals of its formal theory and practical application. Lack of space prevents us from giving here the technical details of computational argumentation and how reasoning or decision making is formalized within such a framework. These can be found, for example, in the tutorial articles in [4]. It is also important to note that the human solver/developer working with LLM-COGNICA does not need to know the underlying technical details of the argumentation reasoner in the same way that s/he does not need to know the technical details of the LLM. It is sufficient to have an understanding of the reasoner at the cognitive level of the COGNICA language.

We will thus briefly outline, in semi-formal terms, the nature of the argumentation-based reasoning with the COGNICA language. Conditionals in this transform naturally into argument schemes [29, 31] of association from conditions as **premises** to a concluding **claim**. For example, the COGNICA statement "**IF** Conds **THEN**

Concl", respectively "**ONLY IF** NConds **THEN** Concl", map into a scheme ASuff, respectively into schemes ANecc, ANPoss, as follows:

(1) ASuff(Concl): (Conds ⇒ Concl)
(2) ANecc(Concl): (not NConds ⇒ not Concl)
(3) ANPoss(Concl): (NConds ⇒ Concl)

Reasoning is then performed under arguments constructed from such argument schemes by grounding these concrete instances of conditions and conclusions. We can then reason to a **possible conclusion** if this can be supported by a "good quality" (set of) argument(s), usually referred to as **acceptable**. Although formally there are different definitions of the property of acceptability, the basic idea is the same and simple: **an acceptable set of arguments defends back against all its counterarguments.** Such an acceptable set of arguments then forms a solid case for the conclusion. But it is also possible that there can be an equally solid case – acceptable argument set – supporting another conflicting conclusion, in which case the argumentation-based reasoning is in a **dilemma**.

For the case of Decision Making a particularly appropriate case of Computational Argumentation, is that of preference-based argumentation [1, 16, 24].COGNICA operates using the Gorgias[17] preference-based argumentation framework as symbolic reasoner. In this a preference or strength relation between arguments regulates the argumentative reasoning, by regulating the defense relation between arguments.In general, an argument can defend against a counterargument only when it is at least as strong as the counterargument. This means that some conclusions cannot be drawn as the arguments supporting them are too weak to defend against counterarguments and hence an acceptable case for the conclusion cannot be drawn.

The strength of arguments is expressed explicitly or implicitly in the decision policy guidelines. For example, in the requests handling policy above, arguments from the fourth and last sentence are stronger than those generated from the first sentence. The first preference comes from a commonsense strength of necessary conditions over sufficient conditions while the second preference comes from an implicitly stated strength of the alternative option to return the requests. Hence when the Decision Policy is translated from Natural Language into the COGNICA language we are aiming to capture, along with the generation of the conditional sentences, the relative preference or strength between them. This is done through the modalities in the COGNICA language, and other canonical strength relations between the type of sentences in COGNICA , as for example:

(1) ANecc(Concl) ≻ ASuff(Concl), for any Concl.
(2) ASuff(Concl1) ≻ APoss(Concl2) when Concl1 and Concl2 are conflicting.

Then in the example problem scenario given above (a request arrives via the helpdesk from a non-collaborator agent) the argument, ANecc(Reject), is stronger than the competing argument ASuff(Accept), and thus it can defend against it but not vice-versa. Hence only an acceptable argument for the option to reject can be build and so we cannot accept the request. If instead, the request came from a collaborating agent, we can build an acceptable argument from the first conditional for accepting the request. But this could be an **unreliable decision** because we could build an equally strong counter-argument against it from the last sentence under the **hypothesis** that it does not have the 'necessary approvals". As a result, the more reliable decision would be to be in a dilemma between accepting or returning, depending on whether the request has the necessary approvals or not. This type of hypothetical reasoning, alongside the other forms of conditional reasoning and modality reasoning, is naturally captured in argumentation by including a universal type of a **hypothetical argument**, with which we can support any statement, without the need for any premise conditions, but which is relatively weaker than any other argument supporting a conflicting opposite statement.

Together with the ability to carry out hypothetical reasoning, another important characteristic of argumentation-based reasoning is that this is naturally **explanative**. Argumentation has a direct connection with explanation [11, 13, 32]. The arguments supporting a conclusion or decision naturally form a justifying explanation for this. The criterion of acceptability for allowing an argument set as a case to derive a (plausible) conclusion or decision forms the basis for extracting an explanation. The premises of the argument(s) supporting conclusion provide the **attributive** part of an explanation, while the defending arguments, within the acceptable set, provide a **contrastive** element of the explanation. These defending arguments contain the reasons that explain why the attributive reasons for the conclusion are indeed at least as strong as the reasons supporting conflicting conclusions.

Hence in the example scenario examined above, where we have a request submitted via the helpdesk by a non-collaborating agent, the conclusion to "reject" which is based on the argument ANecc(Reject), would thus be explained with an attributive part "because it is from a non-collaborating agent". Furthermore, the relative strength of the argument ANecc(Reject) over the argument ASuff(Accept) allows us to further expand the explanation with a contrastive part.

> The decision is "Reject" the request, because it is from a "non-collaborating agent". This reason is stronger than the reason of "submitted via the official helpdesk" supporting the conflicting decision of "Accept".

This is an example of an informative explanation that we can extract from the reasoning process of argumentation and present in some template form expressed in structured natural language:

> "Option 1 holds because of "Reason1". This reason is stronger than "Reason2" for Option 2 when the following "Conditions" hold."

Similarly, for situations where we have a dilemma, we have explanation templates that inform why this is so. Examples of such templates in a structured natural language form are:

> "Option 1 is supported by "Reasons1". Option 2 is also supported by the equally strong "Reasons2"."

> "Whether "Option" is a possible choice or not depends on whether the following "Assumptions" hold or not."

Explanations in such structured form are then easily and accurately transformed into natural language via a simple LLM prompt.

Explanations that depend on assumptions result from the hypothetical reasoning supported by argumentation and play an important role in the framework of LLM-COGNICA . Apart from

safeguarding against missing or incomplete information, assumptions form an **actionable** part of the explanation, indicating actions to be taken to find out whether these assumptions hold or not. For example, in our running example of the requests policy if we do not know whether a request has the necessary approvals, we would get an explanation that points this out. Hence, in an automated agent setting, such tasks of needing to know whether assumptions hold or not could be sent to another agent decision module specialized on this, e.g., the requests policy module would query another module specialized on the approvals compliance of requests.

Argumentation-based reasoning is a very versatile form of reasoning that supports, together with its basic forms of deductive and preference reasoning, other forms of reasoning, such as counterfactual reasoning and exploratory reasoning. We can explore assumptions on problem scenarios but also assumptions on the relative strength of policy guidelines. This allows us to develop algorithms for highly informative explanations and analysis of problems which can be used as the basis for tools that provide feedback to the human designer and developer of an agent decision system.

2.2 Related work

Customizing LLMs for code generation into some programming language is currently under intense investigation [14, 30]. The general aim is to relieve the programmer from the full burden of programming and to facilitate the development of systems directly from their natural specification. The choice of the target programming language can be a procedural one, like Python, or a specialized and declarative language, like that of Answer Set Programming [28].

Our work follows this approach using as a target language the CNL of COGNICA , i.e., a language that is itself a subset of natural language, interpretable by the human problem developer/solver in the same way as the original problem specification in natural language. This "language to language" translation separates it from the other approaches. It allows the CNL of COGNICA to act as a middle ground to evaluate the policy and corrective feedback. Feedback can be given on the translated theory in COGNICA , based on the human solver's reading and reasoning with the translated theory and the argumentation-based explanations generated from this. In this way the programming effort of the problem solver can be reduced to high-level feedback on the Natural Language or CNL expression of the problem. The human problem solver is kept active in the loop with validation and reliability control.

From the symbolic reasoning side of the LLM-COGNICA synthesis, the choice of argumentation-based reasoning, is a natural choice. In the area of autonomous and multi-agent systems [21] argumentation has been used to study different problems, such as goal-decision, agent negotiation, compliance with norms of behaviour, etc. Although the argumentation-based approach gives clear and elegant theoretical solutions to these problems, it is difficult to turn these into practical and scalable solutions. In some cases, the declarative technology of Answer Set Programming, whose underlying theory is closely related to argumentation [10], can be used to develop practical solutions [20], but this is restricted to specific agent application domains. Using LLMs as the natural interface for agent design, development, and deployment, as in the framework of ArgLLMs [12] and our work, offers a way to make the argumentation-based approach to agent problems practically viable.

Our approach can also be viewed as a case of argument mining [19], where, like the work in [5], it employs the natural language processing power of LLMs to mine the policy text. An important difference is that this is not mined directly into formal technical argument structures, but rather it is translated into a CNL policy that can be canonically transformed into argument structures. As above, this means that it is easier to keep the human problem solver actively involved in the development process.

3 FROM NATURAL LANGUAGE TO COGNICA

The COGNICA CNL has been designed to function as an intermediate language between a natural language expression of policies and a symbolic programming language based on an argumentation reasoner. In essence, it is a subset of natural language concerned with expressing conditional sentences, containing several sentence templates for conditional reasoning. Figure 2 shows the current form of the COGNICA language. As presented in Section 2, these sentences give argument schemes together with a preference relation between them stemming from the natural strength of these sentences via their linguistic form, e.g., an ALWAYS modality is stronger than no modality, which is stronger than a MAYBE modality.

1	**NORMALLY** {Conclusion}	Represents a general case of the policy
2	**IF** {Condition} **THEN** {Conclusion}	Represents a simple conditional rule. The Condition is sufficient for the Conclusion.
3	**IF** {Condition} **THEN** {Conclusion} **UNLESS** {Condition1} | **IN SUCH CASES** {Conclusion1}	Condition1 is an exception to the Condition-Conclusion relation. Conclusion 1 is Optional
4	**ONLY IF** {Condition} **THEN MAYBE** {Conclusion}	Condition is necessary for the possibility of the Conclusion, but it's not sufficient.
5	**IF** {Condition} **THEN ALWAYS** {Conclusion}	Represents a more absolute Condition – Conclusion correlation.
6	**IF** {Condition} **THEN MAYBE** {Conclusion}	The Conclusion is possible, but not certain, when a Condition is met.
7	**ONLY IF** {Condition} **THEN** {Conclusion} **UNLESS** {Condition1} | **IN SUCH CASES MAYBE** {Conclusion1}	Condition is necessary Conclusion, but not when Condition1 holds. (Conclusion 1 is Optional.)

Figure 2: The COGNICA Language

The challenge is to develop an embedding mechanism from free natural language into the COGNICA language that preserves the semantic content of the policy. We address this by adapting an LLMs to carry out such a translation, intertwined with tools for the evaluation of the translation that automatically or via the human developer can help improve the translation.

Prompt engineering techniques, including prompt chaining and few-shot prompting, are employed to instruct the LLM to translate a policy. In the prompt to the LLM, comprehensive instructions are provided on the steps to take to achieve the desired translation, alongside with an illustrative example of translation. We give here a brief description - see Appendix A for the full prompt.

The prompt is divided into three sections. In the first part of the prompt, the COGNICA CNL sentence templates were presented and explained, along same lines as in the table of Figure 2. The second part outlines the steps of the algorithm that the LLM must follow to perform the translation. The first step of this is to identify the conclusions supported by each policy sentence, in accordance with the vocabulary of conclusions/options of the policy given by the developer. If a policy sentence uses a word synonymous to an option provided by the developer, the LLM is instructed to use the developer provided word as the supported conclusion. The second step of the algorithm involves the identification of the conditions that support the conclusion of each policy phrase. After this step, the LLM will have extracted "premise-conclusion"associations from each policy sentence. In the third step, the LLM is instructed to select the most appropriate COGNICA CNL phrase template for each such association, using a set of detailed selection guidelines.

The algorithm incorporates secondary processes involving the grammatical and syntactic form of the COGNICA CNL in relation to Natural Language. This relates to the use of negation, conjunction and disjunction so that their basic logical form is mapped appropriately within the constraint form of the COGNICA language. Then, as a third part of the command to LLM, an illustrative example of a policy is provided, accompanied by its translation to COGNICA CNL, before finally, including in the prompt, the actual decision policy to be translated along with the policy's (complementary) options. The prompt is transmitted to the LLM as the user message; no distinction has been made between developer and user messages. In the current version of the LLM-COGNICA system the translator uses OpenAI's o4-mini reasoning model. The model's reasoning effort is set to "medium" to achieve an effective trade-off between reasoning speed and reasoning accuracy.

3.1 Development support in LLM-COGNICA

Within the LLM-COGNICA development environment the translation generated by the LLM can be evaluated both for its semantic consistency with respect to the original policy and for its formal quality with respect to the formal reasoning of argumentation. Tools are provided to assist the developer in identifying elements of improvement of the translated policy as well as providing support to improve the original policy. This feedback and adaptation mechanism is depicted in Figure 3, showing three sources of evaluation of the translated policy: Syntax Verification, Semantic Similarity and Formal Analysis. Feedback from all three sources can be automatic, back to the LLM, or to the human developer who can edit the policy both at the Natural Language level or the translated COGNICA CNL level. We briefly describe these evaluation tools.

The syntax verification process ensures that the policy translation complies with the COGNICA CNL's syntactic restrictions. The process involves two stages. First, the translated policy phrases are verified against COGNICA CNL conditional phrase templates and the CNL syntax restrictions. If any syntax violations are detected, an LLM is used to resolve them. The LLM is given the original policy, the policy translation in COGNICA CNL, the syntactically incorrect phrases and the policy options as input. We instruct the LLM to modify only the phrases with syntax errors. The prompt provided to the LLM includes the COGNICA CNL conditional phrase templates and syntax restrictions regarding the use of conjunction words (the use of 'or' is prohibited, while 'and' is only permitted within phrase conditions). The full prompt for the syntax verification mechanism can be found in Appendix B.

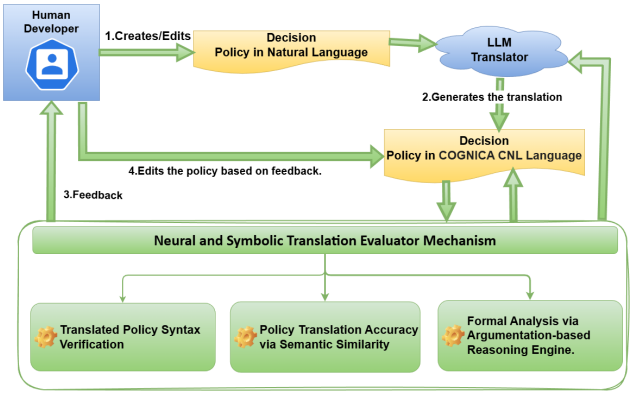

Figure 3: Feedback support in LLM-COGNICA

The extent to which the translation captures the meaning of the original policy, can be examined using text embedding models, such as SBert [27] and the Universal Sentence Encoder [7] between the two forms of the policy. The cosine similarity of these embeddings gives us a measure of their semantic relevance. While high levels of semantic relevance do not guarantee an accurate translation of the policy, low levels indicate a weakness. In instances where the similarity score falls below a threshold, the translation process can be repeated, with the LLM's reasoning effort set to higher levels and the developer can be alerted to examine the translation.

We can use the argumentation-based reasoning engine to formally analyze the translated policy for various properties, such as whether it contains ambiguities, i.e., cases which result in dilemmas or it contains "anomalies", i.e., scenarios with possible incoherent or redundant reasoning. For example, a translated policy containing a pair of sentences of the form "**IF** A **THEN** B **UNLESS** C **IN SUCH CASES** D" and "**IF** A AND C **THEN** not D" is incoherent in whether D is supported in a problem scenario containing A and C. Similarly, in a policy where the same conclusion is supported both under a strong modality and a weak modality would indicate that the weak part of the policy might be redundant. In general, the formal analysis of the translated policy aims to identify parts of the policy that would benefit from a closer investigation. Let us illustrate this support mechanism for the case of our example "requests handling policy" given with its translation in Section 2.

Its formal analysis would identify an ambiguous scenario when "the request is made from a department manager" and " the request lacks necessary approvals" both hold. The reasoning engine's response, as shown in Figure 4, will inform the developer of the ambiguity together with the explanation of why this is so.

The developer thus understands this as a possible non-clear specification and can decide whether the policy needs clarification or not. S/he can then go ahead and edit the policy at the translated level to explicitly resolve the ambiguity. The developer may grant the manager the right to submit a request without the necessary

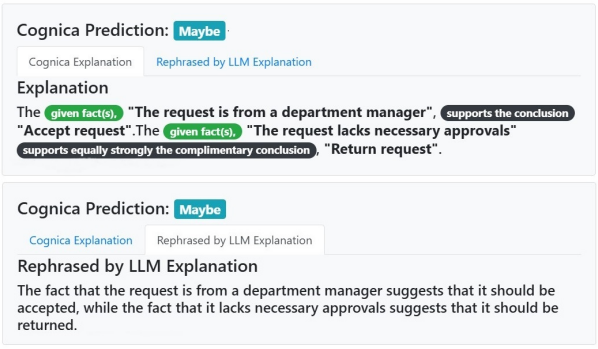

Figure 4: Example of COGNICA support Explanation

approvals by modifying the fifth translated sentence as follows: "**IF** the request lacks necessary approvals **THEN ALWAYS** return request, **UNLESS** the request is from a department manager". Alternatively, the developer may clarify that even the manager must have the necessary approvals and modify the third translation sentence as follows: "**IF** the request is from a department manager and the request has necessary approvals **THEN ALWAYS** accept request". The developer could also choose to extend or modify the original policy and then re-run the translation process. A similar unclear scenario would be that of a request made from a department manager which is not a collaborating agent. Again the explanation provided by the argumentation reasoner plays an important role in helping the developer understand and resolve the possible issue with the policy.

In summary, the developer is supported so as to be the primary evaluator and final judge of the quality of the translation. The developer decides on the faithfulness of the translation and has the option of manually correcting the translation or altering (part of) the original policy text and rerunning the translation process.

4 EVALUATING LLM-COGNICA

The LLM-COGNICA approach depends on the relation between the "understanding" of the decision policy in Natural Language by the LLM and the formal argumentation-based semantics of the COGNICA CNL form of the policy. It is therefore important to have ways to evaluate this relationship and to use this evaluation to guide the continuous development of the framework. We will present here two points of evaluation: (a) the comparison of the reasoning of LLM with the reasoning of COGNICA and (b) the quality of the translation by the LLM of the policy. These will be carried out under a systematic experimentation process using a large dataset of decision policies. The dataset is available upon request.

Dataset Preparation: The evaluation dataset was created using ten state-of-the-art LLMs: Deep Seek V3, Deep Think R1, Gemini 2.0 Flash, Gemini Thinking Experimental, GPT-4o, GPT-4 Turbo, Grok3, and Grok3 Think. In total, 550 policies were generated, utilizing each model to generate 50 decision policies. The use of multiple LLMs allowed us to overcome any potential biases in the set of policies associated with LLMs. The aim was to formulate policies in a variety of domains with varying complexity that would demonstrate different levels of **ambiguity, specificity, and formality**.

The range of the policies extends from policies that utilize one-sentence (usually expressing a general requirement with an exception) template to those that employ multiple CNL sentences templates. As shown in the table of Figure 5 the majority of policies (56%) consist of two to three sentences. A further 23% of policies comprise four to six sentences. Only 18% of these policies consist of a single sentence. The most complex policies are expressed in a series of 7–10 sentences.

Number of Policy Sentences	**1**	**2-3**	**4-6**	**7-10**	**Total**
% Of policies	104 (18.91%)	311 (56.55%)	126 (22.91%)	9 (1.64%)	550
Number Of Options	**1**	2	3	4	5
% Of policies	9.64%	48.36%	38.36%	2.91%	0.73%

Figure 5: Distribution of Policies in Dataset

The complexity of the dataset's policies is also related to the number of complementary options within each policy. The table of Figure 5 shows that the majority of policies (86%) had two or three options, with only 9% involving decision-making for a single option (and its negation). The presence of two or three options in the majority of policies is an appropriate level of complexity for individual module policies regulating the decision making for a particular problem within an overall policy composed of such individual policies.

An important factor in the design and methods of our experiments is the characterization of the policies in terms of the different forms of requirements, in terms of the reasoning that they involve, that it contains. The table in figure 6 shows the distribution of the various forms of reasoning across the different policies in the dataset. As expected, most of the policies contain sufficiency requirements whereas 38% of incorporate involve weak or strong modalities. Furthermore, 38% of the policies include statements expressing exceptions.

Sufficient	Necessary	Exceptions	Strong Modality	Weak Modality	General case
501	29	213	125	85	64
91.09%	5.27%	38.73%	22.73%	15.45%	11.64%

Figure 6: Distribution of Reasoning Forms in Dataset

Each policy is annotated by the combination of the different forms of requirements that it contains. The aim of the experiments is to be able to examine the variation of behaviour, e.g. the quality of translation, across the different forms of requirements.

4.1 Comparing the Reasoning of LLM and COGNICA

It is useful to understand where significant differences between the reasoning of LLM and that of COGNICA might lie so that this can guide us in how to better integrate the LLM with the COGNICA reasoner. For this we can set up experiments as follows: (a) choosing a type of reasoning, (b) choose a subset of policies and a set of queries on each policy that involve the type or reasoning in (a). Then (c) compare the answer of the queries pose directly to LLM on the policies in Natural Language with the answer of COGNICA reasoner

on the translated policies that has been evaluated as faithful. Report (d1) the differences in answers and (d2) qualitative differences in the explanation in all cases irrespective if the answers differ or not. Explanation good quality features, as identified in studies of explanation in cognitive and social science [23], to be used in step (d2) are (e1) non-redundancy of attributive part, (e2) presence of contrastive part and (e3) relevance of attributive part.

There are several types of reasoning in step (a) that can be examined. We will confine ourselves here to present the results of an experiment for the case of **"Reasoning with Ambiguity"** where the ambiguity can be either in the decision policy or it is due to the (incompleteness) of the query scenario.

In one experiment, we have 80 query scenarios in various dataset policies. These are divided into three types of queries:

(a) Scenarios in which a necessary condition is unknown (32.5%).
(b) Scenarios in which an exceptional condition is unknown (33.75%).
(c) Scenarios in which two equally strong sentences (sufficient conditionals) led to different conclusions (33.75%).

The query scenarios were executed on OpenAI's o3-mini reasoning model, applying two levels of reasoning effort: low and high. The same scenarios were also executed in the COGNICA system. Note that the LLM answered the queries using the translated policies expressed in COGNICA CNL as input. This is so that the comparison of answers would depend only on the way the two systems, LLM and COGNICA , reason and not any syntactic differences in the way the policy is expressed.

As we expected, COGNICA 's response to all 80 scenarios was 'Maybe'. When the reasoning effort was medium, the o3-mini responded 40% of the time with "Maybe". This is distributed evenly across all three types of queries. When the reasoning effort increased to high, the percentage of 'Maybe' responses from o3-mini increased only slightly, to 41%. Figure 7 shows the distribution of answers "Yes, No or Maybe" across all queries, for both cases of medium or high reasoning effort. These results clearly show a difference in the reasoning of LLM and COGNICA and hence in the integrated LLM-COGNICA framework we can steer the reasoning of LLMs towards a more formal mode.

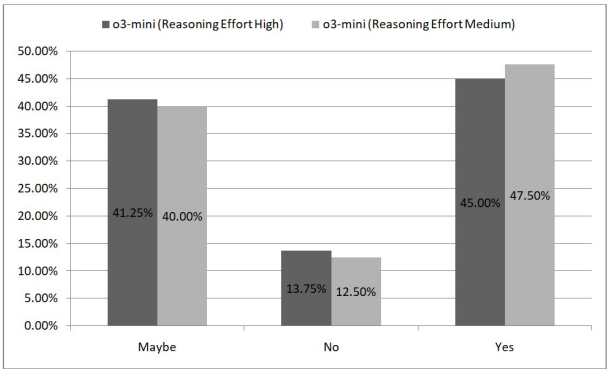

Figure 7: o3-mini responses for reasoning effort medium or high.

At the start of this work we had also compared the COGNICA reasoning with ambiguity with the then early version of CHAGPT 3.5 which in almost all cases did not answer "Maybe" to such dilemma queries. Although we see that the LLM reasoning for this type of queries has improved with the more recent models (and we plan to continuously monitoring the reasoning behaviour of newer LLM models) the fact remains that there is a significant difference in the reasoning and hence this motivates the development of the integration of a symbolic formal reasoner, such as the COGNICA reasoner, with an LLM.

4.2 Evaluating the Quality of the LLM Policy Translation

Each policy in the data set was translated from Natural Language into COGNICA CNL via an LLM prompt analogous to the one presented above in Section 3 (see also Appendix). The results below refer to the use of the O3-mini reasoning model as the translating LLM model. The reasoning effort of this model was set to medium for the translation process.

In order to get a first indication of the semantic faithfulness of the LLM translated policies we carried a first comparison of the policies using the text embedding models of SBERT [27] and the Universal Sentence Encoder [7] between the two forms of the policy. The cosine similarity of these embeddings was computed for each policy and the dataset was partitioned in terms of this in three groups as shown in the table of figure 8.

Cosine Similarity score Group	1-0.9	0.9-0.7	0.7-0.4	Total
SBERT number of policies per Group	183	337	30	550
Number of Policies in Group that require changes	68(37%)	122(36%)	13(43%)	203(37%)
Universal Sim. Encoding (USE) number of policies per Group	34	401	115	550
Number of Policies in Group that require changes	13(38%)	143(36%)	47(41%)	203(37%)

Figure 8: SBERT & USE Semantic Similarity Translation Scores

Subsequently, a manual review of all the translated policies has been conducted. During this a record was kept of the number and type of changes required to make the translated policy closer to the original policy. A total number of 203 (37%) of the policies needed modification. The table of figure 9, shows the number and percentage of policies that needed modification according to the size of the policy. As expected, the larger the policy the more likely is the fact that it might need modification, as it can also be seen in the graph of figure10.

We can also compare the results of this manual review, with the similarity scores, as shown in the table of figure 8. Within the similarity groups as a percentage of the total group size, the percentage of sentences that required modification was distributed evenly at rates close to the translation modification rate for all policies, at approximately 35%-40%. This indicates that the semantic scores is a weak indicator of whether the translation needs modification or not. Nevertheless, within the subgroup of policies needing modification

Size of Policy	Modified Translated Polices	Non-modified Translated Polices	
1-2	76 (26.39%)	212(73.61%)	288
3-4	96(42.67%)	129(57.33%)	225
5-6	22(78.57%)	6(21.43%)	28
7-8	7(100)		7
9-10	2(100)		2

Figure 9: Percentages of Modified Translated Policies

the majority of these policies exhibited a similarity score below 0.9 (66% SBERT and 94% USE).

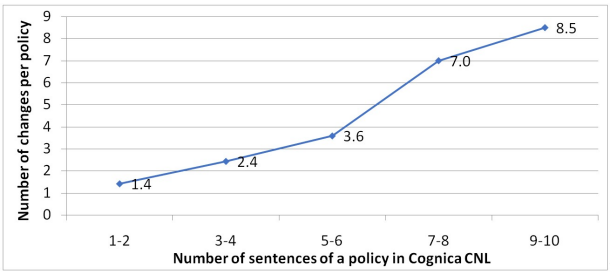

Figure 10: Distribution of Modifications

The table in Figure 11 illustrates the frequency of reasoning form combinations within the dataset in relation to the percentage of policies in a given category that were modified.

Reasoning form combinations	Non-Modified Policy	Modified Policy	Total
Sufficient & Exceptions	133 (76.88%)	40 (23.12%)	173 (31.45%)
Sufficient	88 66.17%	45 (33.83%)	133 (24.18%)
Sufficient & Strong Modality	5 8.77%	52 (91.23%)	57 10.36%
Sufficient & Genaral case	41 89.13%	5 (10.87%)	46 8.(36%)
Sufficient & Weak Modality	23 74.19%	8 (25.81%)	31 5.(64%)
Total	**290**	**150**	**440**

Figure 11: Modifications Across Reasoning Forms

This distribution of needed modification accross different forms of reasoning, indicates that the LLM demonstrates a high degree of accuracy in translating policies involving exceptions in a notable percentage of cases. On the other hand, there is a clear difficulty in recognizing modality especially the strong modality captured by "ALWAYS" conditionals. In 91% of cases where the 'Sufficient & Strong Modality' combination is present, the policy has been modified to accurately reflect the original one.

In the COGNICA CNL language, strong conditionals are employed to strengthen a statement; however when there are no explicit linguistic markers indicating the strength of a sentence but rather the strength is implicit across several sentences the LLM fails to recognize that implicit strength, resulting in incorrect usage of 'always'. In general, the experiment indicates that the translation can benefit from more precise guidance to the LLM on the relative strength that a sentence gains through the incorporation of modality words.We intend to fine-tune an LLM using a training dataset involving policies with multiple forms of reasoning and multi-phrase translations to address these issues. As mentioned above in section 3, this can be a continuous process of evaluation of the translated policies by the human developers and feedback to the LLM for improving/adapting its translation of policies.

5 CONCLUSIONS AND FUTURE WORK

We have presented a neuro-symbolic framework that integrates the Natural Language capabilities of LLMs with explainable and reliable argumentation-based reasoning. The choice of Argumentation as a symbolic formal reasoner was motivated by its close connection to human reasoning, allowing the problem solving it to be adaptive in the face of a continuous flow of changing and often contradictory information from the external (problem) environment.

The proposed integrated framework forms the central element of a platform, whose current form is sufficiently mature for the development of real-life agent systems. The platform has been used to develop a cognitive medical assistant based on the official medical guidelines for diagnosis and treatment of Schwannomatosis[26]. This has been positively received by medical practitioners and patients and it is currently undergoing an in a real-life setting with clinical experts in the disease.

The challenge of reliability of the systems developed under the synthesis of LLMs with the COGNICA argumentation-based reasoner can be addressed in various ways. We can improve the translation from Natural Language to the COGNICA language either by further training the LLM translator and/or extending the COGNICA language with new sentence templates that are needed to capture requirements that are found to be difficult to translate. We are planning to fine-tune an LLM using a dataset of policies relating to cases in which the LLM performed poorly during the translation evaluation experiment.We can also extend the formal analysis of the translated policies, resulting in further support tools for the human developer in evaluating and building their systems. Furthermore, translated policies can be modularized into levels facilitating its scrutiny for reliability and correctness with the requirements.

Within the LLM-COGNICA framework the solutions of problems are found via the reasoning with the COGNICA translated policy. Hence, their reliability with respect to this translated policy is absolute, based on the formal guarantees and properties of the argumentation-based reasoning. Once such property is the fact that argumentation handles inconsistent or incomplete information by recognizing such situations as dilemmas and characterizing them in its explanations. This can then be used to give active control to the human developer to adapt the translated policy and/or to clarify the original problem requirements in the policy in Natural Language. The distinguishing feature of our approach, that the whole process of development and problem solving is carried out entirely within Natural Language, facilitates this active role by the human developer.

Learning to develop systems in LLM-COGNICA is a matter of learning to specify problem requirements in natural language in a way that ensures a clear and faithful translation into the COGNICA language, or indeed, learning to work directly in the machine readable language of COGNICA .

REFERENCES

[1] Leila Amgoud and Claudette Cayrol. 1998. On the acceptability of arguments in preference-based argumentation. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence* (Madison, Wisconsin) *(UAI'98)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1–7.

[2] Vaishak Belle, Michael Fisher, Alessandra Russo, Ekaterina Komendantskaya, and Alistair Nottle. 2024. Neuro-Symbolic AI + Agent Systems: A First Reflection on Trends, Opportunities and Challenges. In *Autonomous Agents and Multiagent Systems. Best and Visionary Papers*, Francesco Amigoni and Arunesh Sinha (Eds.). Springer Nature Switzerland, 180–200.

[3] T.J.M. Bench-Capon and Paul E. Dunne. 2007. Argumentation in artificial intelligence. *Artificial Intelligence* 171, 10 (2007), 619–641.

[4] Philippe Besnard, Alejandro Garcia, Anthony Hunter, Sanjay Modgil, Henry Prakken, Guillermo Simari, and Francesca Toni. 2014. Tutorials on Structured Argumentation. *Argument & Computation* 5, 1 (2014), 1–117.

[5] Elfia Bezou-Vrakatseli, Oana Cocarascu, and Sanjay Modgil. 2025. Can Large Language Models Understand Argument Schemes?. In *Findings of the Association for Computational Linguistics: ACL 2025*, Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (Eds.). Association for Computational Linguistics, Vienna, Austria, 13666–13681.

[6] Benjamin Callewaert, Simon Vandevelde, and Joost Vennekens. 2025. VERUS-LM: a Versatile Framework for Combining LLMs with Symbolic Reasoning. arXiv:2501.14540 [cs.AI] https://arxiv.org/abs/2501.14540

[7] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Eduardo Blanco and Wei Lu (Eds.). Association for Computational Linguistics, 169–174.

[8] E. Dietz and A. Kakas. 2021. Cognitive argumentation and the selection task. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, Vol. 43. 1588–1594.

[9] E. Dietz, A. Kakas, and A. Koumi. 2022. Cognitive Machine Argumentation. In *Proceedings of the 4th European Conference on Argumentation (Studies in Logic & Argumentation)*. College Publications.

[10] Uwe Egly, Sarah Alice Gaggl, and Stefan Woltran. 2010. Answer-set programming encodings for argumentation frameworks. *Argument & Computation* 1, 2 (2010), 147–177.

[11] Débora Engelmann, Juliana Damasio, Alison R. Panisson, Viviana Mascardi, and Rafael H. Bordini. 2022. Argumentation as a Method for Explainable AI : A Systematic Literature Review. In *2022 17th Iberian Conference on Information Systems and Technologies (CISTI)*. 1–6.

[12] Gabriel Freedman, Adam Dejl, Deniz Gorur, Xiang Yin, Antonio Rago, and Francesca Toni. 2025. Argumentative Large Language Models for Explainable and Contestable Claim Verification. *Proceedings of the AAAI Conference on Artificial Intelligence* 39, 14 (2025), 14930–14939.

[13] Yihang Guo, Tianyuan Yu, Liang Bai, Jun Tang, Yirun Ruan, and Yun Zhou. 2023. Argumentative Explanation for Deep Learning: A Survey. In *2023 IEEE International Conference on Unmanned Systems (ICUS)*. 1738–1743.

[14] Juyong Jiang, Fan Wang, Jiasi Shen, Sungju Kim, and Sunghun Kim. 2025. A Survey on Large Language Models for Code Generation. *ACM Trans. Softw. Eng. Methodol.* (2025).

[15] P.N. Johnson-Laird and R.M.J. Byrne. 2002. Conditionals: A theory of meaning, pragmatics, and inference. *Psychological Review* 109(4) (2002), 646–678.

[16] Souhila Kaci and Leendert van der Torre. 2008. Preference-based argumentation: Arguments supporting multiple values. *International Journal of Approximate Reasoning* 48, 3 (2008), 730–751.

[17] Antonis C. Kakas, Pavlos Moraitis, and Nikolaos I. Spanoudakis. 2018. Gorgias: Applying Argumentation. *Argument and Computation* 10, 1 (2018), 55–81. https://doi.org/10.3233/aac-181006

[18] R.A. Kowalski. 1990. English as a logic programming language. *New Generation Computing* 8(2) (1990), 91–93.

[19] John Lawrence and Chris Reed. 2019. Argument Mining: A Survey. *Computational Linguistics* 45, 4 (Dec. 2019), 765–818.

[20] Nicola Leone and Francesco Ricca. 2015. *Answer Set Programming: A Tour from the Basics to Advanced Development Tools and Industrial Applications*. Springer International Publishing, Cham, 308–326.

[21] Nicolas Maudet, Simon Parsons, and Iyad Rahwan. 2007. Argumentation in Multi-Agent Systems: Context and Recent Developments. In *Argumentation in Multi-Agent Systems*, Nicolas Maudet, Simon Parsons, and Iyad Rahwan (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 1–16.

[22] Hugo Mercier and Dan Sperber. 2011. Why do humans reason? Arguments for an argumentative theory. *Behavioral and Brain Sciences* 34, 2 (2011), 57–74.

[23] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence* 267 (2019), 1–38. https://doi.org/10.1016/j.artint.2018.07.007

[24] Sanjay Modgil. 2009. Reasoning about Preferences in Argumentation Frameworks. *Artificial Intelligence* 173, 9 (2009), 901 – 934.

[25] Liangming Pan, Alon Albalak, Xinyi Wang, and William Wang. 2023. Logic-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 3806–3824.

[26] Melpo Pittara, Anastasia Kyriacou, Adamos Koumi, Maria Matsangidou, Eirini Schiza, Constantinos S. Pattichis, and Antonis Kakas. 2025. An Explainable AI System for Clinical Decision Support in Schwannomatosis. In *Artificial Intelligence Applications and Innovations. AIAI 2025 IFIP WG 12.5 International Workshops*, Antonios Papaleonidas, Elias Pimenidis, Harris Papadopoulos, and Ioannis Chochliouros (Eds.). Springer Nature Switzerland, Cham, 25–37.

[27] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv:1908.10084 [cs.CL] https://arxiv.org/abs/1908.10084

[28] Manuel Borroto Santana, Irfan Kareem, and Francesco Ricca. 2024. Towards automatic composition of ASP programs from natural language specifications. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence* (Jeju, Korea) *(IJCAI '24)*. Article 685, 9 pages.

[29] S.E. Toumlin. 1958. *The Uses of Argument*. Cambridge University Press.

[30] Lukas Twist, Jie M. Zhang, Mark Harman, Don Syme, Joost Noppen, and Detlef D. Nauck. 2025. LLMs Love Python: A Study of LLMs' Bias for Programming Languages and Libraries. *CoRR* abs/2503.17181 (2025). https://doi.org/10.48550/arXiv.2503.17181

[31] Douglas N. Walton. 1996. *Argumentation Schemes for Presumptive Reasoning*.

[32] Kristijonas Čyras, Antonio Rago, Emanuele Albini, Pietro Baroni, and Francesca Toni. 2021. Argumentative XAI: A Survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Zhi-Hua Zhou (Ed.). International Joint Conferences on Artificial Intelligence Organization, 4392–4399.

A LLM PROMPT FOR TRANSLATION OF A DECISION POLICY TO COGNICA CNL.

```
You are an expert in Cognica CNL, controlled natural languages, and decision policy translation.
Your task is to translate natural language decision policies into valid Cognica CNL statements,
following the rules below exactly.

You must be precise, conservative, and strictly compliant.

[1] Cognica CNL – Allowed Templates

Cognica CNL includes the following 7 templates:

1. NORMALLY {Conclusion}.
   Definition: Represents the general scenario of the decision policy when no condition applies.
   The Conclusion outlines the action to be taken in this scenario.

2. IF {Condition} THEN {Conclusion}
   Definition: Used when a specific outcome or consequence is definite if the condition is met.

3. IF {Condition} THEN MAYBE {Conclusion}
   Definition: Used when the outcome is uncertain upon meeting the condition, implying a degree of probability.

4. IF {Condition} THEN ALWAYS {Conclusion}
  Definition: Indicates a preferred / dominant rule: when the condition is satisfied,the conclusion is strongly favored
   and is intended to override competing weaker rules

5. IF {Condition} THEN {Conclusion} UNLESS {Condition2} IN SUCH CASES {Conclusion2}
   Definition: A more complex template that incorporates an exception. Generally, if the first condition is met,
   the corresponding conclusion follows; however, if the second condition also applies, a different outcome occurs.

6. ONLY IF {Condition} THEN MAYBE {Conclusion}
   Definition: Indicates that the condition is necessary for the possibility of the conclusion,
   but it is not sufficient to guarantee it.

7.ONLY IF {Condition1} THEN {Conclusion1} UNLESS {Condition2} IN SUCH CASES MAYBE {Conclusion1}
  Definition: The condition is necessary for the possibility of the conclusion, but not sufficient.
  An exception to the necessity is introduced by exceptional condition.

Note: You must use one of these templates exclusively; combining them or creating new templates is not allowed.
Also, the uppercase keywords shown in the templates must be used as indicated.

[2] Translation Procedure (MANDATORY)

To translate a given decision policy into Cognica CNL, follow these steps for each policy sentence:

Step 1: Identify Supported Conclusions
1. Determine Supported Conclusions:
   Identify the conflicting conclusions that each phrase supports. (A sentence may support one or more conclusions.)

2. Handle Unlisted Conclusions:
   If you encounter a conclusion that is not in the provided list of conflicting conclusions, proceed as follows:
   - 1.2.1: If the identified word has an opposite in the list, use the negation of that opposite.
            Example: If you identify “deny call” and the list includes “allow call,” then use “NOT allow call.”
   - 1.2.2: If the identified word has a synonym in the list, use that synonymous word.
   - 1.2.3: If the conclusion includes a precondition, remove the precondition and use a synonym of the remaining conclusion
   from the list.
```

The removed precondition must then be incorporated as a condition in the translation.
Example: For “allow call from manager” (with precondition “call is from manager”),
the conclusion becomes “allow call.”

Step 2: Identify Supporting Conditions
For each identified conclusion, determine the conditions that support it.
(Note that a sentence may support a conclusion without any preceding conditions.)

Step 3: Express the Translation Using Templates
Express each identified conclusion together with its supporting conditions by following these guidelines:

1. No Conditions:
   If a conclusion is not supported by any conditions, express it using the template:
   NORMALLY {Conclusion}

2. Conditions Present (Without Exception):
  - If the conditions are necessary for the possibility of the conclusion but not sufficient to guarantee it, use Template 6:
     ONLY IF {Condition} THEN MAYBE {Conclusion}
   - If there are conditions with no exceptions, choose one of the basic templates (Templates 2, 3, or 4)
   based on the certainty of the outcome:
       - Possible but Uncertain Outcome:
         IF {Condition} THEN MAYBE {Conclusion}
       - Certain Outcome:
         IF {Condition} THEN {Conclusion}
       - Preferred Conclusion / dominant rules:
         IF {Condition} THEN ALWAYS {Conclusion}

3. Conditions with Exception:
   If there are conditions that are necessary but not sufficient, and an exception applies, use Template 7:
    ONLY IF {Condition1} THEN {Conclusion1} UNLESS {Condition2} IN SUCH CASES MAYBE {Conclusion2}
4. Exceptions in Basic Templates:
   If exceptions to the condition–conclusion relationship are detected, use Template 5:
   IF {Condition} THEN {Conclusion} UNLESS {Condition2} IN SUCH CASES {Conclusion2}

Step 4: Rephrase Conjunctions
If any translation from Step 3 contains conjunctions (e.g., “or”, “and”) within conditions or conclusions,
split the translation into multiple sentences instead of using these conjunctions.
The word ‘AND’ may only be used to link conditions when the conclusion arises from their combination.
AND = not allowed to combine multiple conclusions, or rules.
Example:
    Incorrect example (violates the rule)
        IF the customer has a coupon AND it is the weekend THEN apply discount AND send promotional email.
    Correct version (split into separate sentences)
        IF the customer has a coupon AND it is the weekend THEN apply discount.
        IF the customer has a coupon AND it is the weekend THEN send promotional email.

OR is not allowed anywhere in Cognica CNL.
    Incorrect example (violates the rule)
        IF the user is an admin OR the user is a manager THEN allow access.
    Correct version (split into separate sentences)
        IF the user is an admin THEN allow access.
        IF the user is a manager THEN allow access.

Step 5: Rephrase Negations
For any condition or conclusion that includes negation via contractions, rephrase it using “not” instead of the contraction.
    Example1: Replace “didn't go” with “not go.”

```
    Example2: Replace “does not go” with “not go.”

Step 6. Ensure SWI-Prolog Compatibility
   Replace any characters that are syntactically incompatible with SWI-Prolog syntax in the conditions and conclusions
    of the sentences.
    Example:
    “10% of people” → “ten percent of people”

#Translation Examples:

Policy:
""Authenticated users are granted access to the website.
Unauthenticated users are redirected to login unless accessing public pages or holding a guest link.
Banned users are always denied access.
During maintenance or high traffic, only admins are granted access; others are denied or shown a maintenance page.""

Policy Options:"Grant Access,Deny Access,Redirect to Login,Show Maintenance Message"

Expected response:
IF user is authenticated THEN Grant Access.
IF user is unauthenticated THEN Redirect to Login UNLESS user is holding guest link IN SUCH CASES Grant Access.
IF user is banned THEN ALWAYS Deny Access.
ONLY IF system is NOT under maintenance THEN Grant Access UNLESS user is admin IN SUCH CASES MAYBE Grant Access.
ONLY IF system is NOT experiencing high traffic THEN Grant Access UNLESS user is admin IN SUCH CASES MAYBE Grant Access.
IF system is under maintenance THEN     Deny Access UNLESS user is admin IN SUCH CASES Grant Access.
IF system is experiencing high traffic THEN Show Maintenance Message UNLESS user is admin IN SUCH CASES Grant Access.

policy:"{{policy}}"
policy_options:"{{policy_options}}"]}
```

B LLM PROMPT FOR SYNTAX VERIFICATION MECHANISM.

```
You are a Cognica CNL guided syntactic repair engine.

Your task is to FIX a set of Cognica CNL rules by correcting ONLY the
syntactic mistakes explicitly provided as input.

You MUST preserve the intent of the ORIGINAL POLICY.
You MUST NOT invent new policy content.
You MUST NOT fix issues that are NOT listed in the provided mistakes.


1 CANONICAL COGNICA CNL TEMPLATES

You are allowed to use ONLY the following 7 templates.
You MUST use exactly ONE template per rule.
You MUST NOT modify these templates.

1. NORMALLY {Conclusion}.

2. IF {Condition} THEN {Conclusion}.

3. IF {Condition} THEN MAYBE {Conclusion}.

4. IF {Condition} THEN ALWAYS {Conclusion}.

5. IF {Condition1} THEN {Conclusion1} UNLESS {Condition2} IN SUCH CASES {Conclusion2}.
```

6. ONLY IF {Condition} THEN MAYBE {Conclusion}.

7. ONLY IF {Condition1} THEN {Conclusion1} UNLESS {Condition2} IN SUCH CASES MAYBE {Conclusion2}.

Uppercase keywords are mandatory.
No other surface forms are permitted.

2 NON-NEGOTIABLE COGNICA CNL RULES

- Use ONLY the templates above
- Exactly one template per rule
- AND is forbidden in conclusions
- OR is forbidden everywhere
- AND is allowed in conditions only if all conditions are jointly required
- Negation must use explicit "not"
- Conclusions must belong to POLICY OPTIONS (or valid negations)
- Output must be syntactically valid Cognica CNL

3 GUIDED REPAIR CONSTRAINT

You are given a list of SYNTACTIC MISTAKES.
You MUST:
- Fix ALL listed mistakes
- Fix ONLY the listed mistakes
- Apply the MINIMAL change required for each fix

If a listed mistake requires splitting a rule into multiple rules,
you MUST do so.

4 OUTPUT FORMAT (HARD STOP)

- Output ONLY the fixed Cognica CNL rules
- One rule per line
- No explanations
- No comments
- No blank lines

5 INPUT

ORIGINAL POLICY:
{{original_policy}}

COGNICA CNL RULES:
{{rules}}

SYNTACTIC MISTAKES:
{{syntactic_mistakes}}

POLICY OPTIONS:
{{policy_options}}