Scalable, Context-Aware NLP Moderation for Child Safety: A Multi-Agent Ethical and Legal Compliance Framework

Jan Fillies[1,4,*], Theodoros Mitsikas[2,4], Ralph Schäfermeier[3,4] and Adrian Paschke[1,4,5]

[1]Freie Universität Berlin, Germany
[2]National Technical University of Athens, Greece
[3]Leipzig University, Germany
[4]Institute of Applied Informatics, Germany
[5]Fraunhofer FOKUS, Germany

Abstract

Protecting children from harmful online content requires systems that are accurate, adaptive, and legally compliant across jurisdictions. This paper presents a hybrid, rule-based, multi-agent moderation architecture designed to detect and mitigate toxic speech in real time while ensuring adherence to diverse legal and ethical standards. The system employs large language models, including Google Gemini, GPT-4o-nano, and GPT-4o, to classify user messages according to a detailed hate speech taxonomy. In addition to use case specifically defined ethical rules the approach dynamically identifies the applicable legal frameworks (e.g., COPPA, GDPR, DSA) based on the participants' country of origin and uses LLM-driven agents to generate relevant legal obligations as executable rules in the Prolog rule language. This is the base for a legal and ethical reasoning agent. Moderation decisions are thus context-sensitive, policy-aligned, and legally grounded. System performance was evaluated on a human-annotated dataset of illegal hate speech, demonstrating its effectiveness in identifying content that violates legal definitions. By integrating unsupervised classification with symbolic rule-based reasoning, the system offers a scalable, reliable solution for protecting children and others in online communication environments.

Keywords

Symbolic and Sub-Symbolic AI, Multi-Agent System, Content Moderation, Hate Speech Detection

1. Introduction

Children increasingly inhabit dynamic digital spaces, such as social media, gaming chats, and educational forums, that require complex, real-time content moderation [1]. Moderation must go beyond detecting toxic language to address overlapping legal frameworks, cultural sensitivities, and ethical obligations, all while safeguarding privacy and platform autonomy.

Conventional methods, reliant on static filters or opaque models, lack the transparency, adaptability, and legal interpretability needed to enforce nuanced laws like COPPA, GDPR, or the Digital Services Act. Legal definitions, such as that of "illegal hate speech", vary by jurisdiction, creating a need for systems that adapt to linguistic and regulatory diversity. Platform owners must be able to select applicable legal regimes, but translating these into formal, actionable rules remains a significant challenge.

This paper presents a hybrid AI architecture that integrates large language models (LLMs) with symbolic reasoning to enable compliant, real-time moderation of children's online interactions. The system uses LLMs, including Google Gemini and GPT-4o, paired with a detailed taxonomy of harmful content to support accurate classification. These classifications are mapped to obligations from relevant legal frameworks. Through a novel Rule Generation API, LLM agents generate symbolic inputs that follow the Prolog syntax, powering a legal reasoning agent implemented in Prova (a rule language that combines Prolog with Java) [**?**], that assesses content in context and provides justifiable moderation

RuleML+RR'25: Companion Proceedings of the 9th International Joint Conference on Rules and Reasoning, September 22–24, 2025, Istanbul, Turkiye

*Corresponding author.
jan.fillies@fu-berlin.de (J. Fillies); mitsikas@central.ntua.gr (T. Mitsikas); ralph.schaefermeier@imise.uni-leipzig.de (R. Schäfermeier); adrian.paschke@fu-berlin.de (A. Paschke)
0000-0002-7570-3603 (T. Mitsikas); 0000-0002-4349-6726 (R. Schäfermeier); 0000-0003-3156-9040 (A. Paschke)

outcomes. Evaluation on a human-annotated dataset of illegal hate speech shows the system's ability to produce accurate and explainable rule-based decisions.

The key contributions of this work are:

1. A multi-agent moderation architecture that integrates LLM-based content evaluation with symbolic legal reasoning;
2. an automated rule generation mechanism that translates legal obligations into Prolog clauses using LLM agents;
3. an empirical validation of the system's ability to detect illegal hate speech using real-world annotated data.

The prototype is available at Discord[1] and the code can be found at GitHub[2].

2. Related Research

Content moderation has emerged as a critical concern in the era of digital communication, particularly for safeguarding vulnerable populations such as children. Traditional moderation techniques, often reliant on keyword filters or simple machine learning classifiers, have been criticized for their lack of contextual understanding and legal interpretability [2, 3]. Recent advances in natural language processing (NLP), particularly transformer-based language models like BERT [4] and GPT [5], have significantly improved the detection of toxic and hateful content. However, their black-box nature poses challenges for transparency, auditability, and regulatory compliance—key requirements in child safety contexts.

Efforts to introduce rule-based reasoning into moderation systems have gained momentum as a way to address these limitations. Hybrid approaches that combine neural models with symbolic reasoning have shown promise in increasing both interpretability and accuracy [6, 7]. In particular, multi-agent systems have been used to simulate human-like decision-making in complex environments, making them well-suited for context-sensitive moderation tasks [8].

From a legal perspective, compliance with child protection laws such as COPPA in the United States, GDPR in the EU, and more recently, the Digital Services Act (DSA), imposes stringent obligations on platform operators. These legal instruments require proactive measures to detect and remove illegal content while ensuring due process and privacy protection [9]. Yet, most current moderation systems do not support customizable legal reasoning, leaving a gap in enforceable compliance mechanisms.

Recent research has explored automated rule generation using NLP techniques to encode policy and legal standards as executable logic [10], but integration with live moderation pipelines remains limited. In this context, recent work by Fillies et al. integrates legal and ethical reasoning into moderation pipelines using multi-agent architectures [11]. Their GDPR-compliant chat application restricts harmful content when minors are present and generates personalized counter speech. They don't allow for dynamic alignment of moderation actions with platform-specific legal frameworks. The proposed approach extends this line of work by enabling automated legal rule generation, LLM-based classification, and real-time rule evaluation within a multi-agent architecture.

3. System Design

The system (see Figure 1) enables dynamic, autonomous, rule-based moderation of chat content using LLM-based agents that detect, generate, and apply formalized legal and ethical rules. It consists of two core processes: one for clause generation and another for content moderation. Each process leverages semantic rule technologies and agent-based reasoning to ensure compliance with platform-specific policies. A top-level agent dynamically determines whether new facts need to be generated or if existing facts sufficiently cover the country of origin of the users sending messages on the chat platform.

[1] https://discord.gg/9fSZJZSd
[2] https://github.com/fillies/GuardianAgents/tree/main

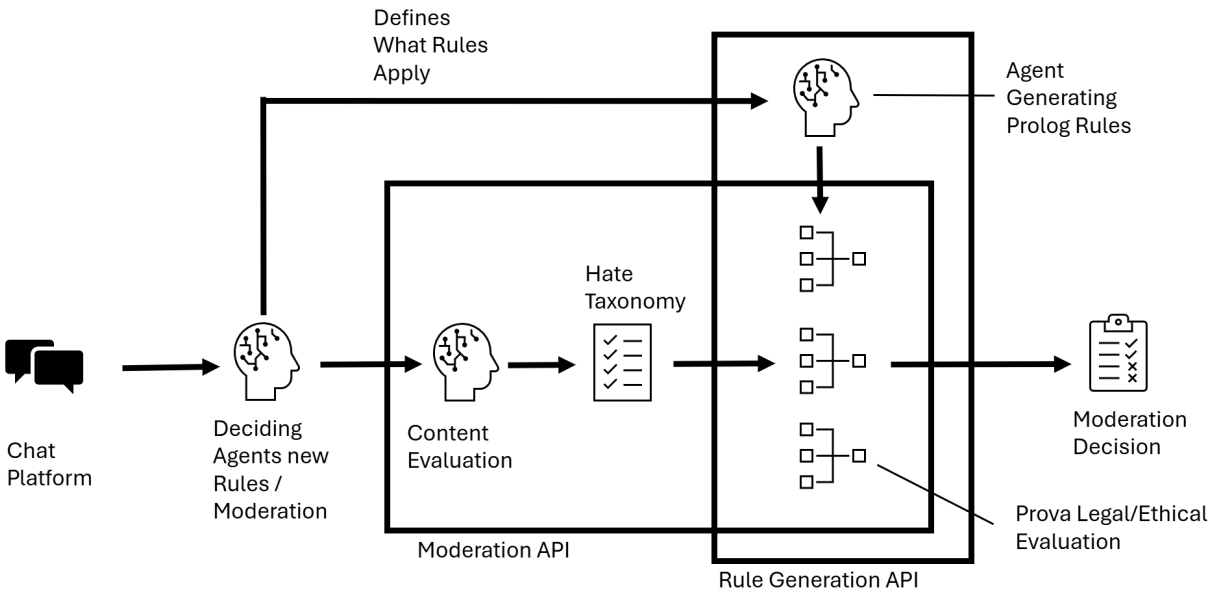

Figure 1: System Overview

3.1. Rule Generation Flow

The Rule Generation Process encodes the platform owner's moderation policies into machine-readable rules using Prova, a rule-based semantic scripting language designed for agent environments [**?** 12]. Prova was chosen due to its logical inference agility, paired with its event-driven character and its seamless integration with Java, which enables direct invocation of Java methods and facilitates interaction with external systems and libraries, which together make it optimal for the moderation use case.

Whenever a user from a previously unrepresented region writes in the moderated chat platform, the normative framework—such as applicable legal, ethical, or community guidelines—is selected by an agent and translated into executable Prova facts. The resulting rules are semantically enriched and hierarchically organized, allowing for modular reuse and logical inference. A Legal/Ethical agent validates and applies these formalized rules, exposing them for downstream use and ensuring traceability and adaptability in dynamic regulatory contexts.

3.2. Moderation Flow

The Moderation Process applies the generated clauses in real time to evaluate user-generated content. Incoming messages from a chat platform are first processed by an LLM and agent-based content evaluation module, which categorizes content using a large and granular predefined hate taxonomy. This classification is then passed to a reasoning engine that applies the Prova rules to determine whether the selected content adheres to platform policies. The output is a moderation decision, which is both explainable and enforceable. This architecture supports compliance, transparency, and scalability in automated content governance.

4. Prototype

4.1. Agent Setup

The prototype is implemented using the graphical AI agent workflow builder Langflow[3]. Langflow, an open-source framework, was chosen for its ability to facilitate rapid prototyping and transparent

[3]https://github.com/langflow-ai/langflow

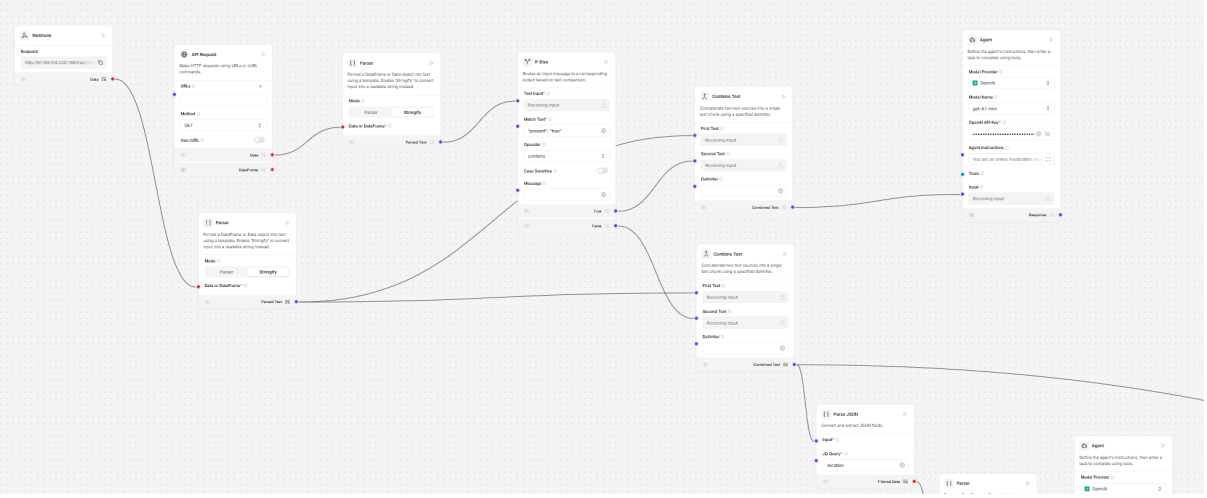

Figure 2: High level overview of the Agent Flow in Langflow, see GitHub for the enlarged original Image

orchestration of agent interactions. The model (in Langflow terms referred to as *flow*) coordinates the conditional execution of each agent, the tools they use, and the data transfer between the components involved, including the submission of potential content moderation messages to the chat system.

Figure 2 shows an extract of the agent pipeline in Langflow.

The execution of an instance of the flow is triggered via a webhook notification initiated by the chat system whenever a new message is posted by a user. The notification message includes the chat message itself, the chat message's id, and the user's location.

```
{
  "message_id": "54934",
  "location": "california",
  "message": "Hi everyone, my name is Bob"
}
```

In order to decide whether Prova clauses for the given location need to be created, a call to Prova's API location check endpoint is made, which returns a JSON object indicating either the presence or the absence of a rule set for the location in question. The flow includes a conditional branch, depending on the content of Prova's response. In case the location has not been encountered before and Prova reports the absence of corresponding facts, the branch leads to a call to the legal agent, which gathers the relevant legal frameworks for the given location and outputs them in the following format, which serves as the input format for the next agent.

```
[
  {
    "name": "Full name of the law or regulation",
    "citation": "Specific citation, e.g., §130 StGB or Article 8 GDPR",
    "jurisdiction": "National / Regional / International",
    "year_enacted": "Year of enactment or most recent version",
    "description": "Brief description of what the law regulates",
  },
  ...
]
```

The Prova agent is sequentially connected to the legal agent and works on the latter's output. Its prompt commands it to create Prolog facts in a format accepted by the Prova engine.

```
[
  "isIllegal(public_safety, dangerous_conspiracy, violence_or_threats, stochastic_terrorism,
  ↪  usa ,\"Illegal under 18 U.S.C. §371\").",
```

```
  "isIllegal(sexual_or_obscene, explicit_minor, _, _,usa, \"Illegal under 18 U.S.C.
  ↪  §2252A\").",
  "isIllegal(sexual_or_obscene, nonconsensual, _, _, usa, \"Illegal under 18 U.S.C.
  ↪  §2241\").",
  "isIllegal(sexual_or_obscene, sexual_harassment, _, _, usa, \"Illegal under 18 U.S.C.
  ↪  §2244A\").",
]
```

The resulting array of clauses along with their associated location are sent to the Prova rule upload endpoint.

The branch then remerges with the main branch and continues with the hate speech classifying agent, which accepts the incoming message and classifies it according to our taxonomy. Its output format is as follows.

```
{
  "message_id": "msg_10475",
  "location": "california",
  "content": "You're so mature for your age. Don't tell anyone we talk like this :wink:. I
  ↪  saw your new pic—hot. If you don't send more, I'll have to show your messages to your
  ↪  parents. And don't bother blocking me, I'll make sure your school finds out what you've
  ↪  been doing.",
  "tags": {
    "sexual_or_obscene": ["sexual_harassment"],
    "harassment": ["repeated_abuse", "reputational_threats"],
    "coercion_or_grooming": [
      "manipulation",
      "normalization",
      "emotional_blackmail"
    ]
  }
}
```

The output, including the classification result and the originating location, is sent to the Prova legal and ethical rule checker. Depending on the legal rules' validity in the given location, the output of the checker includes all violations identified as such, accompanied with explaining references to the respective laws.

The result is then sent back to the chat system, where it is used for generating an explanatory content moderation message.

4.2. Legal and Ethical Evaluation with Prova

The Prova-based [12] legal and ethical evaluation agent accepts two input types. The first input type consists of the Prolog facts that are generated. These are parsed and asserted to the Prova rulebase, enriching it over time. The implementation does not include any hardcoded legislation, instead expecting to be generated and asserted. On the other hand, the ethical evaluation is hardcoded at the current development stage, to be expanded in the future to include custom ethical rules.

It accepts two kinds of facts. The first are simple facts, as in the following example:

```
isIllegal(sexual_or_obscene,sexual_harassment,_,_,usa,"Illegal under 18 U.S.C. §2244A").
```

Simple facts capture legislation that is sufficiently represented with a single pair of category-sublabel, as seen in the first two arguments. In the above example, sexual harassment is illegal in the specific jurisdiction, without requiring any other conditions to be applied, thus the next pair of arguments are anonymous variables. Finally, the last two arguments are the jurisdiction and some explanatory text for the legal basis.

The second kind of accepted facts are facts with "conditional combinations", as seen below:

```
isIllegal(hate_speech,extremist_symbols,violence_or_threats,symbolic_threat,california,
↪  "Illegal under CA Penal Code § 11411").
```

For example, in California, symbolic threats or displaying extremist symbols are not necessarily illegal on their own. However, this is not the case when a symbolic threat is made using an extremist symbol.

The second input type pertains to the message evaluation. In particular, each message and its characteristics is evaluated against the asserted facts, by reactive rules.

```
legalChecker() :-
    rcvMult(X,P,F,complianceRequest,[Cat,ListSubCat,Region]),
    illegalOrCombination(X,Cat,ListSubCat,Region,Law),
    spawn(X,$Service,resume,[]).
```

While the input, after preprocessing, is relayed to Prova using its message passing capabilities, the result is communicated utilizing Prova's capability to call external methods (via `spawn/4`) [12], constructing this way the JSON response. This is performed through the auxiliary predicate `illegalOrCombination/5` which is implemented as follows:

```
illegalOrCombination(X,Cat1,ListSubCat1,Region,Law) :-
    element(SubCat1,ListSubCat1),
    isIllegal(Cat1,SubCat1,Cat2,SubCat2,Region,Law),
    bound(Cat2), % conditional combinations
    spawn(X,$Service,triggerHasComplexRules,[]).

illegalOrCombination(X,Cat1,ListSubCat1,Region,Law) :-
    element(SubCat1,ListSubCat1),
    isIllegal(Cat1,SubCat1,Cat2,SubCat2,Region,Law),
    free(Cat2), % simple fact
    spawn(X,$Service,updateResponse,[X,"legal_violation",Cat1,SubCat1,Law,Region]).
```

The above implementation handles simple facts, and also is able to determine if conditional combinations are needed, depending on if the third argument of `isIllegal` is a free variable.

As an optimization, this is performed only if rules that require combinations are applicable to the user's jurisdiction, i.e., if the `triggerHasComplexRules()` Java method has been invoked. This is because in such case, each pairwise combination of tags should be evaluated by Prova, which is computationally intensive. The rule responsible for handling conditional combinations considers only rules without free variables:

```
legalChecker() :-
    rcvMult(X,P,F,complianceRequest,[Cat1,ListSubCat1,Cat2,ListSubCat2,Region]),
    element(SubCat1,ListSubCat1),
    isIllegal(Cat1,SubCat1,Cat2Ground,SubCatGround,Region,Law),
    bound(Cat2Ground),
    equal(CatGround,Cat2),
    ...
    spawn(X,$Service,updateResponse,[X,"legal_violation",Cat1,SubCat1,Law,Region]),
    spawn(X,$Service,updateResponse,[X,"legal_violation",Cat2,SubCat2,Law,Region]),
    spawn(X,$Service,resume,[]).
```

Regarding ethical evaluation, at the current development stage, it is performed through a single reactive rule which considers hardcoded ethical facts:

```
ethicalChecker() :-
    rcvMult(X,P,F,complianceRequest,[Cat1,ListSubCat1,Region]),
    element(SubCat1,ListSubCat1),
    isUnethical(Cat1,SubCat1),
    spawn(X,$Service,updateResponse,[X,"ethical_violation",Cat1,SubCat1,Cat1,Region]),
    spawn(X,$Service,resume,[]).
```

4.3. LLM based Clause Generation

The system uses a two-stage pipeline powered by large language models to generate executable legal rules that govern moderation decisions. This process is implemented using two Langflow agents.

The first agent, operating on GPT-4o-nano, is responsible for identifying all relevant legal frameworks based on a user's administrative region, such as a country or state. GPT-4o-nano was chosen for the first stage because its efficiency and low computational cost make it well suited for structured retrieval at scale. When provided with a region, the agent returns a structured JSON array containing the full name of each applicable law, its legal citation (such as Article 8 of the GDPR or §130 of the German Criminal Code), the jurisdictional scope (national, regional, or international), the year of enactment or last revision, and a brief description of its regulatory scope. This ensures that the legal basis for moderation is both specific and regionally appropriate.

The second Langflow agent uses GPT-4o to convert the retrieved legal frameworks into executable Prolog-style clauses, based on a predefined taxonomy of problematic content. This taxonomy includes categories such as `hate_speech`, `violence_or_threats`, `public_safety`, and others, each with associated sublabels (e.g., `misgendering`, `true_threat`, `school_targeting`).

For each label, and combinations where legally significant, the agent generates facts in the following format:

```
"isIllegal(meta_label, label, conditional_meta, conditional_label, jurisdiction,
↪  "legal_basis")"
```

The agent adheres to strict constraints: only labels from the taxonomy are used; rules must be based on known legal frameworks; and conditional combinations (e.g., incitement + extremist symbols) are explicitly modeled.

The output is returned as a raw JSON array of rule strings and is directly ingested by the moderation engine's legal reasoning agent. This enables automated, traceable, and jurisdiction-aware enforcement of moderation policies in real time.

The full prompts and agents can be seen on GitHub[4].

4.4. LLM based Problematic Speech Detection

To classify user-generated messages according to potential policy or legal violations, the system employs a lightweight Gemini 2.0 Flash-Lite model embedded in a Langflow agent. The Gemini 2.0 Flash-Lite model was selected for this stage because its lightweight architecture provides rapid, cost-efficient classification while maintaining sufficient accuracy for content moderation tasks. This agent is tasked with identifying harmful content based on a predefined taxonomy (See Appendix A) that covers a wide spectrum of online abuse, including hate speech, threats, harassment, coercion, and policy violations. The classification process plays a foundational role in enabling both ethical and legal reasoning, as it determines whether content matches any regulated or prohibited categories prior to rule-based evaluation.

The agent operates by receiving a message object, including a unique identifier and the text content, and returning a structured JSON output that annotates the content with relevant labels. These labels are drawn exclusively from a predefined multi-level taxonomy, which includes top-level categories such as `hate_speech`, `violence_or_threats`, `sexual_or_obscene`, and `coercion_or_grooming`, each with a detailed list of subcategories. For example, if a message exhibits manipulative behavior toward a minor, the agent may apply tags such as `emotional_blackmail` under `coercion_or_grooming`, or `reputational_threat` under `harassment`, depending on the context of the language used.

The classification agent is designed to operate deterministically, without prompting follow-up questions or generating open-ended explanations. It returns only the required JSON structure, ensuring consistency and machine-readability across the moderation pipeline. The result is a precise, multi-label content annotation that informs downstream agents in the system—most notably, the legal reasoning engine responsible for determining rule violations under applicable jurisdictional frameworks.

By tightly coupling the output to the taxonomy and formatting it as structured data, this approach enables real-time classification that is both scalable and transparent. The use of GPT-4o-mini balances

[4]https://github.com/fillies/GuardianAgents/

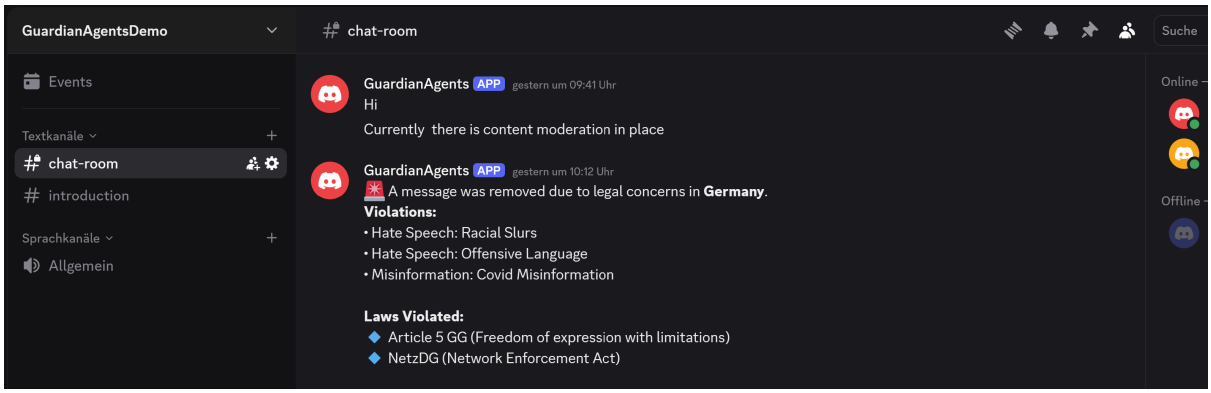

Figure 3: Prototype Chat Room on Discord

model efficiency with semantic accuracy, making it suitable for live deployment in chat environments where latency and explainability are critical.

The full prompt and agent can be seen on GitHub[5].

4.5. Prototype Chat Room on Discord

To demonstrate the practical application of the moderation framework, a live prototype was developed and integrated into a Discord server. This setup allows for real-time content evaluation and moderation decisions based on the legal and ethical rule engine described in previous sections.

The chat room is designed with onboarding and moderation workflows that reflect both platform policies and regulatory compliance. When a new user joins the server, they are assigned a temporary "Quarantine" role and prompted to introduce themselves in a specific format, stating their name, country of origin, and age. This initial message is parsed by the system to associate the user with a region, which is then used to select the applicable legal framework for content moderation.

Messages sent in the main chat channel are intercepted and evaluated through a webhook that forwards the content and associated region to the Langflow pipeline. The pipeline classifies the message based on the speech taxonomy and assesses legal violations through LLM-based classification and rule-based reasoning. If violations are detected, the message is programmatically deleted using the Discord API. In such cases, a summary message is posted in the channel that lists the legal violations and the specific laws that were breached, providing users with a transparent explanation of the moderation action. An example of such a moderated message is shown in Figure 1.

The system also includes administrative tools: when new rules are generated by the rule generation agent, an automated direct message is sent to the server administrator containing the rule set, message context, and regional information. This ensures both traceability and manual oversight where needed.

This integration showcases the operational feasibility of the proposed architecture in a real-time, user-facing environment. It highlights how agent-based legal reasoning, combined with LLM-powered classification, can be deployed to enforce region-specific moderation policies in a scalable and explainable manner.

5. Evaluation

Regarding the generated Prolog clauses, the LLM-based generation did not produce invalid syntax, and also handled well the anonymous variables (which are present in the majority of facts). Similarly, the message classification JSON output (which is subsequently passed to the Prova) was also unproblematic with respect to the syntax validity, and also adhered to the predefined taxonomy of problematic content.

[5] https://github.com/fillies/GuardianAgents/tree/main/langflowAgents

We evaluated the system's ability to generate legal rules and assess hate speech in accordance with European Union legal standards, using a legally grounded dataset of 158 annotated posts[6], developed by Zufall et al. (2022) [13]. This dataset aligns with the EU Framework Decision 2008/913/JHA and contains social media messages labeled for legal punishability. Of the 158 messages, 24 were classified as punishable, while the remaining 134 were considered highly toxic but not illegal under the EU Framework Decision 2008/913/JHA.

To also evaluate performance on non-toxic inputs, the 134 non-punishable but toxic messages were paired with 134 completely non-toxic messages sourced from [14], which contains real-life, non-toxic chat messages exchanged between teenagers on Discord.

All messages were presented to the application as if they were real-time communications sent within the EU. The system's outputs were recorded and compared to the dataset's ground-truth labels. Precision, recall, and F1 score were used as evaluation metrics.

5.1. Results

The evaluation focused on three classes: (1) Toxic but not illegal, (2) Toxic and illegal (punishable), and (3) Non-toxic and non-illegal.

As shown in Table 1 and Table 2, the system performed well at identifying non-toxic, non-illegal content (Class 3), achieving high precision (1.000), recall (0.926), and an F1 score of 0.962. However, it failed to recognize highly toxic but non-punishable messages (Class 1), scoring 0 across all metrics. This suggests the system struggles to distinguish socially harmful but legal speech from other categories, possibly due to the fact that during the training of LLMs, legal but toxic statements are also against the policy.

For legally punishable content (Class 2), recall was perfect (1.000), but precision was low (0.143), leading to many false positives. While the system successfully identified all punishable cases, it frequently misclassified non-punishable content as illegal.

Averaged metrics show moderate overall performance, with a macro F1 score of 0.404 and a micro F1 of 0.507. The high weighted recall (0.902) indicates a conservative bias, prioritizing detection of potentially illegal content at the expense of precision.

Class	Precision	Recall	F1 Score
Class 1	0.000	0.000	0.000
Class 2	0.143	1.000	0.250
Class 3	1.000	0.926	0.962

Table 1
Per-class metrics

Average	Precision	Recall	F1 Score
Macro	0.381	0.642	0.404
Micro	0.507	0.507	0.507
Weighted	0.475	0.902	0.461

Table 2
Averaged metrics of all three classes combined

6. Discussion

The evaluation highlights both the strengths and limitations of our hybrid moderation system. While the system performs well at identifying non-toxic content and detecting legally punishable speech with high recall, it tends to over-classify borderline toxic content as illegal, resulting in a significant number of false positives. This conservative bias is build with intention in a system designed for child protection, where caution is preferred. However, it also underscores the difficulty of translating complex, nuanced legal standards into executable rules without overgeneralization.

A key insight is that rule generation based on jurisdiction-specific legislation is possible but remains incomplete. The problem of automatic rule generation is in no way solved, but this system offers a first step when human-based rule generation is not possible due to time and cost constraints. It will be important to further evaluate the rules at the case level to gain a deeper understanding of the

[6]https://github.com/simulacrum6/op-hate-nlp/tree/main

ability of LLM-based rule generation. Furthermore, certain forms of hate speech, such as subtle or coded expressions, may evade detection if they are not explicitly covered in the legal framework or not captured by the current taxonomy. This limitation highlights the need for continuous refinement of both the hate speech taxonomy and the legal rule generation pipeline.

Given that the system is designed for environments with minors, privacy and data security are critical concerns. The handling of sensitive or potentially harmful content must comply with regulations such as GDPR. Storing or processing user-generated content, especially toxic or illegal material, poses risks related to inadvertent exposure, misuse, or unauthorized access. In the current prototype, content handling does not yet incorporate advanced measures for protecting personally identifiable information, but also does not explicitly save any, future scaling to larger platforms will require rigorous encryption, access control, and data anonymization strategies.

The system's reliance on LLMs for classification and rule synthesis introduces several risks that are particularly relevant in the context of child safety and legal compliance. LLMs can hallucinate, misinterpret legal standards, or misalign culturally, leading to both false positives and false negatives. While the system incorporates conservative thresholds and rule-based safeguards, these do not fully eliminate risks. Critically, legal misinterpretation could result in either over-censorship, which impacts free expression, or under-censorship, which compromises child safety. Therefore, human oversight and formal legal validation mechanisms remain necessary to ensure accountability, reliability, and trustworthiness at scale.

Finally, while the Discord-based prototype demonstrates feasibility in a real-life environment, scaling this approach to larger platforms will require optimization of latency, handling of adversarial inputs.

7. Conclusion and Future Work

This paper introduced a scalable, multi-agent moderation framework that integrates LLM-based classification with symbolic legal and ethical reasoning to ensure context-aware and jurisdiction-specific content moderation. By combining automated rule generation with real-time detection, the system addresses key challenges in protecting children from harmful online content while aligning with diverse legal standards.

Evaluation results show the system's strong ability to detect illegal content, though precision remains a limitation, particularly in distinguishing toxic but lawful speech. This trade-off reflects a deliberate, risk-averse stance in child protection scenarios, but points to the need for more nuanced moderation capabilities.

Future work will address this by refining the hate speech taxonomy, improving rule specificity, and incorporating more robust validation mechanisms, potentially involving legal experts or structured legal ontologies. Additionally, we aim to extend the system's language coverage, handle adversarial or obfuscated content, and deploy it in more diverse, high-volume communication environments. This will support broader applicability across global platforms while maintaining transparency and legal compliance.

Acknowledgments

The 'Terminology and Ontology- Based Phenotyping (TOP)' project is funded by the German Federal Ministry of Education and Research (grant number: 01ZZ2018).

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT in order to: Grammar and spelling check, and reword. After using this service, the authors reviewed and edited the content as needed and takes full responsibility for the publication's content.

References

[1] Y. Theocharis, S. Kosmidis, J. Zilinsky, F. Quint, F. Pradel, CONTENT WARNING: Public Attitudes on Content Moderation and Freedom of Expression, https://doi.org/10.17605/OSF.IO/F56BH, 2025.

[2] T. Gillespie, Custodians of the Internet: Platforms, Content Moderation, and the Hidden Decisions That Shape Social Media, Yale University Press, 2018.

[3] E. Chandrasekharan, C. Gandhi, M. W. Mustelier, E. Gilbert, Crossmod: A Cross-Community Learning-Based System to Assist Reddit Moderators, Proceedings of the ACM on Human-Computer Interaction 3 (2019) 1–30.

[4] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding, in: 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), 2019, pp. 4171–4186.

[5] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language Models Are Few-Shot Learners, Advances in neural information processing systems 33 (2020) 1877–1901.

[6] C. Liang, J. Berant, Q. Le, K. D. Forbus, N. Lao, Neural Symbolic Machines: Learning Semantic Parsers on Freebase With Weak Supervision, arXiv preprint arXiv:1611.00020 (2016).

[7] T. R. Besold, S. Bader, H. Bowman, P. Domingos, P. Hitzler, K.-U. Kühnberger, L. C. Lamb, P. M. V. Lima, L. de Penning, G. Pinkas, et al., Neural-Symbolic Learning and Reasoning: A Survey and Interpretation, in: Neuro-Symbolic Artificial Intelligence: The State of the Art, IOS press, 2021, pp. 1–51.

[8] M. Wooldridge, An Introduction to Multiagent Systems, John Wiley & Sons, 2009.

[9] C. Novelli, F. Casolari, P. Hacker, G. Spedicato, L. Floridi, Generative ai in eu law: Liability, privacy, intellectual property, and cybersecurity, Computer Law & Security Review 55 (2024) 106066. URL: https://www.sciencedirect.com/science/article/pii/S0267364924001328. doi:https://doi.org/10.1016/j.clsr.2024.106066.

[10] I. Chalkidis, A. Jana, D. Hartung, M. Bommarito, I. Androutsopoulos, D. M. Katz, N. Aletras, LexGLUE: A Benchmark Dataset for Legal Language Understanding in English, arXiv preprint arXiv:2110.00976 (2021).

[11] J. Fillies, T. Mitsikas, R. Schäfermeier, A. Paschke, Agent-Based Hate Speech Moderation Approach, in: International Workshop on Causality, Agents and Large Models, Springer, 2024, pp. 107–125.

[12] A. Kozlenkov, Prova Rule Language version 3.0 User's Guide, 2010. https://github.com/prova/prova/tree/master/doc.

[13] F. Zufall, M. Hamacher, K. Kloppenborg, T. Zesch, A Legal Approach to Hate Speech – Operationalizing the EU's Legal Framework against the Expression of Hatred as an NLP Task, in: N. Aletras, I. Chalkidis, L. Barrett, C. Goanță, D. Preoțiuc-Pietro (Eds.), Proceedings of the Natural Legal Language Processing Workshop 2022, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Hybrid), 2022, pp. 53–64. URL: https://aclanthology.org/2022.nllp-1.5/. doi:10.18653/v1/2022.nllp-1.5.

[14] J. Fillies, S. Peikert, A. Paschke, Hateful Messages: A Conversational Data Set of Hate Speech Produced by Adolescents on Discord, in: International Data Science Conference, Springer, 2023, pp. 37–44.

A. Toxic Language Schema

The content moderation schema is displayed in a structured JSON format for categorizing various types of harmful or policy-violating content. A complete listing of the schema, including tag categories, is provided below.

```
{
  "type": "object",
```

```
  "properties": {
    "message_id": { "type": "string" },
    "content": { "type": "string" },
    "tags": {
      "type": "object",
      "properties": {
        "hate_speech": {
          "type": "array",
          "items": {
            "type": "string",
            "enum": [
              "dehumanization",
              "religion",
              "ethnicity",
              "nationality",
              "gender",
              "sexual_orientation",
              "age",
              "disability",
              "socioeconomic_class",
              "holocaust_denial",
              "extremist_symbols",
              "religious_mockery",
              "misgendering",
              "mocking_disability"
            ]
          }
        },
        "violence_or_threats": {
          "type": "array",
          "items": {
            "type": "string",
            "enum": [
              "direct_threat",
              "true_threat",
              "call_for_violence",
              "incitement",
              "glorification_of_violence",
              "doxxing",
              "stochastic_terrorism",
              "symbolic_threat"
            ]
          }
        },
        "public_safety": {
          "type": "array",
          "items": {
            "type": "string",
            "enum": [
              "dangerous_conspiracy",
              "panic_inducing_claims",
              "disobedience_encouragement",
              "school_targeting"
            ]
          }
        },
        "sexual_or_obscene": {
          "type": "array",
          "items": {
            "type": "string",
            "enum": [
              "explicit_minor",
              "nonconsensual",
              "sexual_harassment",
```

```
            "offensive_language"
          ]
        }
      },
      "legal_or_policy_violation": {
        "type": "array",
        "items": {
          "type": "string",
          "enum": [
            "copyright",
            "defamation",
            "impersonation",
            "spam",
            "political_campaigning",
            "misinformation",
            "voter_suppression",
            "tos_violation"
          ]
        }
      },
      "context_disruption": {
        "type": "array",
        "items": {
          "type": "string",
          "enum": [
            "off_topic",
            "trolling",
            "external_irrelevant",
            "external_harmful",
            "external_misleading"
          ]
        }
      },
      "harassment": {
        "type": "array",
        "items": {
          "type": "string",
          "enum": [
            "repeated_abuse",
            "brigading",
            "reputational_threats",
            "identity_based_harassment",
            "deepfake_impersonation"
          ]
        }
      },
      "self_harm": {
        "type": "array",
        "items": {
          "type": "string",
          "enum": [
            "promotion",
            "glorification",
            "mocking"
          ]
        }
      },
      "coercion_or_grooming": {
        "type": "array",
        "items": {
          "type": "string",
          "enum": [
            "manipulation",
            "normalization",
```

```
                "emotional_blackmail"
              ]
            }
          }
        },
        "additionalProperties": false
      }
    },
    "required": ["message_id", "content", "tags"],
    "additionalProperties": false
}
```