Semantic Web 0 (2023) 1–0
IOS Press

Engineering User-centered Explanations to Query Answers in Ontology-driven Socio-technical Systems

Juan Carlos L. Teze [a], Jose Nicolas Paredes [b], Maria Vanina Martinez [c,*] and Gerardo Ignacio Simari [b]

[a] *Facultad de Ciencias de la Administracion & Consejo Nacional de Investigaciones Cientificas y Tecnicas (CONICET), Universidad Nacional de Entre Rios (UNER), Argentina*
E-mail: carlos.teze@uner.edu.ar

[b] *Departamento de Ciencias e Ingenieria de la Computacion, Universidad Nacional del Sur (UNS) & Instituto de Ciencias e Ingenieria de la Computacion (UNS–CONICET), Argentina*
E-mails: jose.paredes@cs.uns.edu.ar, gis@cs.uns.edu.ar

[c] *Departamento de Computacion, Universidad de Buenos Aires (UBA) & Instituto de Ciencias de la Computacion (ICC UBA–CONICET), Argentina*
E-mail: mvmartinez@dc.uba.ar

Abstract. The role of explanations in intelligent systems has in the last few years entered the spotlight as AI-based solutions appear in an ever-growing set of applications. Though data-driven (or machine learning) techniques are often used as examples of how opaque (also called black box) approaches can lead to problems such as bias and general lack of explainability and interpretability, in reality these features are difficult to tame in general, even for approaches that are based on tools typically considered to be more amenable, like knowledge-based formalisms. In this paper, we continue a line of research and development towards building tools that facilitate the implementation of explainable and interpretable hybrid intelligent socio-technical systems, focusing on features that users can leverage to build explanations to their queries. In particular, we present the implementation of a recently-proposed application framework (and make available its source code) for developing such systems, and explore user-centered mechanisms for building explanations based both on the *kinds* of explanations required (such as counterfactual, contextual, etc.) and the *inputs* used for building them (coming from various sources, such as the knowledge base and lower-level data-driven modules). In order to validate our approach, we develop two use cases, one as a running example for detecting hate speech in social platforms and the other as an extension that also contemplates cyberbullying scenarios.

Keywords: Ontological Languages, Socio-Technical Systems, Explainable Artificial Intelligence, Hate Speech in Social Platforms

1. Introduction

Sharing data through a wide range of social platforms has now become the norm, and such platforms are an example of one particular class of socio-technical system [45]. Even though these systems are very useful tools to connect people socially (as well as in other settings like commerce and work, among others), they are also notoriously vulnerable; it is now commonplace to see attacks taking many forms, such as cyberbullying, hate speech,

*Corresponding author. E-mail: gis@cs.uns.edu.ar.

fake news, and other forms of misinformation and malicious content. For example, recent research [67] showed that hate speech is an increasingly problematic issue that has affected many people and communities that are targets of insult, offense, and even violent actions during the COVID-19 pandemic, turning social platforms into one of the main scenarios where hate speech propagates. This situation has triggered a growing interest in the development of intelligent tools for cybersecurity tasks, and one common requirement is that such tools should have the capacity to offer explanations for their outputs in order to foster a human-in-the-loop scheme that enables better automated decisions. In this context, users of AI-based tools seek to both make effective use and understand how the system is reaching its conclusions, interacting with the system in a collaborative manner. Interestingly, this kind of transparency was already identified at least a decade ago, well before the latest explainability and interpretability boom mostly originated in the machine learning community, as a key feature to ensure that actors take a reflective stance towards the socio-technical systems they interact with [30].

Explainable Artificial Intelligence (XAI) has been mainly addressed from two perspectives: adaptation of traditional formalisms and development of novel approaches. For instance, explanations in symbolic reasoning systems typically provide information about why a conclusion is obtained from a knowledge base, usually encoded as a set of symbolic rules, and an inference procedure. In these approaches, explanatory information often involves descriptions of the reasoning steps (including data and rules) followed by the system to make a certain decision. On the other hand, while some sub-symbolic reasoning systems (also known as machine learning or data-driven models) can be considered explainable by design, such as decision trees and decision rules, the majority of these models work as black boxes and do not reveal sufficient details about their internal behavior. In these cases, explainability typically focuses on exploring the behavior of such opaque models, such as identifying the input features that are most associated with different outputs. Finally, hybrid systems combine symbolic and sub-symbolic reasoning, seeking to both process large amounts of data and build predictive models, as well as deploy rich representations of domain knowledge for higher-level structured reasoning tasks.

Recent works [20, 46] point out that explainability serves and addresses different user-specific purposes, and is adopted in a wide of range of domains. Intelligent systems supporting tasks in the domain of cybersecurity, where users require far more information from the system than a simple output, are a good example. There are several motivations for integrating cybersecurity systems with XAI, chiefly among which is the difficulty that security experts find in understanding the logic behind (and making decisions with) outputs generated by intelligent tools in a security breach or threat assessment. AI tools sometimes provide inaccurate outputs in terms of false positives that mislead experts, compromising the integrity of the entire system. Due to the opaque nature of data-driven AI models, they typically are unable to give detailed information about events of interest. Explainability is thus a feature that enables informed and auditable decision making; XAI-based solutions therefore have the potential for a wide variety of applications not only in cybersecurity but also in other domains that share its characteristics. For example, detecting fraudulent operations and data breaches can be effectively addressed with XAI methods due to their capacity for detailed reasoning over complex patterns. Furthermore, the ability of XAI for providing detailed real-time explanations of processes to assist in final decision-making and adding understandable interpretations makes security systems more resilient to various threats.

Current approaches to XAI in cybersecurity domains focus mainly on specific problems that leverage ad hoc knowledge sources, and thus do not wholly address the explainability problem in a general sense. In this work, we focus on the challenging task of developing the capacity to deliver a range of different kinds of explanations, which points to the need for providing *explanations as a service* via a general framework that interacts with multiple AI models. Our aim is to provide guidelines especially geared towards the design and implementation of high-quality and timely explanations in the context of socio-technical intelligent systems (using cybersecurity applications as examples of such systems), encapsulating components necessary to compose hybrid explanation types that address different goals and expose different forms of knowledge.

Contributions. The contributions of this work can be summarized as follows:

(i) We describe how the HEIC Application Framework[1] (recently proposed up to the design level in [50, 51], here generalized to other domains beyond cybersecurity and renamed to HEIST – *Hybrid Explainable and Intrepretable Socio-Technical systems*), can be used to guide the development of a system called NETDER-HS for detecting hate speech in online social platforms.
(ii) We explore and analyze seven different kinds of primitive explanations, that are well-known in the literature, in the context of AI systems that leverage ontological knowledge as well as network knowledge (which refers to the parts of the KB used in the execution of network diffusion-driven simulations).
(iii) We characterize the explanations presented above in the context of the HEIST framework, and show how they can be applied in the NETDER-HS system.
(iv) We develop a "build-your-own"-style approach for users to choose explanations according to their requirements. Furthermore, we analyze the 21 possible ways in which the primitive explanation styles can be combined in pairs, discussing the interactions and constraints that arise in these combined styles.
(v) We make available a preliminary version of the implementation of the HEIST application framework and document its use, focusing on the design aspects necessary to build explanations following the basic styles studied in this work.

The article is structured as follows: Section 2 provides a brief overview of various issues related to explainability in socio-technical systems, focusing on cybersecurity as an example domain, and introduce the HEIST framework for XAI-based socio-technical systems; Section 3 develops a use case that arises from the instantiation of HEIST; in Section 4, we explore how different kinds of explanations in the context of HEIST can be built, and present a second use case to further illustrate these concepts; Section 5 provides further details by documenting the design and implementation of explainable systems based on our framework; and finally, Sections 6 and 8 describe the related work and conclusions, respectively.

2. Background on Explainability in Intelligent Socio-technical Systems

Explainability has been identified as a key factor for adoption of AI in a wide range of contexts, as the increasing deployment of intelligent systems in different domains has shown that when decisions are taken by these systems, it is essential for practical and social reasons that explanations are available to users. There has been an explosion in the number of works in the literature exploring XAI; in order to have the possibility to provide an adequate background, here we will focus on XAI in cybersecurity, which can be seen as a sub-domain of social-technical systems that includes many challenging aspects and is therefore a representative domain to work with. We begin with the work of [69], which discusses several questions that should receive special attention to tackle the challenges of XAI-based cybersecurity. Then, we present the kinds of explanations that we later contemplate in our proposal. Finally, we briefly review the application framework for XAI-based cybersecurity systems we recently proposed in [50]; here, we generalize it to other domains beyond cybersecurity and rename it to HEIST – *Hybrid Explainable and Intrepretable Socio-Technical systems*.

2.1. Explainable Cybersecurity: Questions Guiding R&D

In [69], the authors highlight several research directions by describing different aspects related to explanations in the cybersecurity context such as *who* gives/receives the explanations, *what* constitutes a good explanation, *when* the explanation should be given, *how* to give explanations, *where* the explanations should be made available, and *why* the explanations are needed. We now briefly discuss each of these questions.

Who? – As we have mentioned, cybersecurity is closely related to socio-technical systems because it combines human efforts with those of more traditional information systems. On the human side, participants occupy roles that

[1] In software engineering, *application frameworks* [62] are general-purpose software tools (such as certain libraries and abstract architectures) designed to support the production of domain-specific applications that are created by *instantiating* (or extending) the framework. Well-known examples of this kind of tool are Angular and React in the domain of web applications.

range from novice, barely computer-literate users, to experts who can write their own code. In particular, malicious actors also fall somewhere in this range, since nowadays there are tools that allow to launch attacks with very little knowledge. Thus, the level of expertise of the user plays an important role in providing an explanation.

What? – Another key related question is to identify the object to be explained; in this context, the level of detail will depend on the user's goals. For example, novice users will likely need explanations that improve their trust in system answers, or that help them to understand how to use the system correctly, whereas further details will be necessary for power users. So, different components will need to be explained, such as automated mechanisms, policies, threat models, vulnerabilities, and countermeasures.

Where? – Location is another issue that needs to be addressed in order to decide where it is best for explanations to be made available. Possibilities include different levels, as part of specific workflows, detached from system functionalities, or incorporating dedicated modules to handle features related to explanations.

When? – Explanations can be provided at various instances, such as before, during, or after a task is carried out. Explainability may be essential for the system to begin functioning (for instance, to explain aspects of input data), it may be crucial to support users' trust at runtime, or the goal may be to support future decisions, making it important for explanations to be available after the execution.

Why? – One of the most important questions regarding explainability is the reason why we need an explanation, since it points to the importance of the users' understanding of the way the system produces its outputs. Some examples include increasing confidence, trust, transparency, usability, verifiability, or testability. Moreover, the importance and need for explanations generally fall into two classes: critical and beneficial. In the former, the system's outputs will not be accepted without adequate explanation, while in the latter such explanations are beneficial, but not crucial.

How? – Finally, a delivery style suitable for the intended audience must be chosen. Styles can be categorized according to language, including:

- *natural language* to produce possibly informal text or sentences that follow a template-based approach;
- *graphical language* to leverage the intuitions of visual components such as explanation trees, graphs, message-sequence charts or statistical histograms;
- *formal language* including code, proofs, and the system's internal representations; and
- *games* to teach users how to interact with the system.

2.2. Explanations in Intelligent Socio-Technical Systems

As we began to argue in the previous section, a key requirement for the success and practical adoption of Intelligent Socio-Technical Systems (ISTSs, for short) is that users must have confidence in the system's outputs and automated decisions. In this sense, automatically generated explanations have been recognized as a fundamental mechanism to increase user trust in AI applications. With this motivation, here we will analyze several kinds of explanations that have been proposed in the literature and are now well-established, and that will be used in the rest of the paper as basic styles. For a more comprehensive review of the literature on XAI, we refer the reader to [31, 41, 64].

Contrastive: Refers to approaches where explanations are based on a strategy that focuses on establishing some type of contrast [23, 44, 54, 64] seeking to answer the question "why was one decision made instead of another". The contrast is then presented between the resulting decision and one or more alternative ones. The goal of this approach is to provide users with sufficiently comprehensive information on the reasoning that led to the system's outputs by answering questions focusing on possible alternatives that may be more intuitive than the ones received.

Counterfactual: As a sub-class of contrastive, counterfactual explanations identify the required changes on the input side that would have significant impact on the output [14, 42, 44]. People are used to considering counterfactuals in daily life; for example, adding new information to incomplete knowledge, making assumptions to evaluate possible scenarios, etc. Counterfactuals have become relevant to a variety of AI applications (*e.g.,* improve training in generative adversarial networks (GANs)), and especially in XAI; the strategy has been particularly successful in sys-

tems that do not rely on data-driven techniques [8]. Although this explanation type is often seen as human-friendly, one drawback that has been identified [5] is the "Rashomon effect", where each counterfactual explanation tells a different (but still true) story to reach a prediction.

Justification-based: In the area of ontological reasoning, there has been a significant amount of work on methods and techniques for computing and working with *justifications* as a dominant and popular form of explanation [35]. In this context, a justification for an entailment is a minimal subset of the ontology that is sufficient for the entailment to hold. The set of axioms corresponding to the justification is minimal in the sense that if an axiom is removed from the set, the remaining axioms no longer support the entailment. Advantages to this approach include, for example, that they are conceptually simple, they have a clear relationship with the underlying ontology, and there are simple presentation strategies that work well in many settings.

Data-driven: This kind of explanations have recently received much attention because they are very useful in complex AI data-driven models [1, 4], which in general seek to derive a mathematical function such that the input is a set of values for features or attributes of an entity, and the output is the value of a particular feature of that entity. Typically, this function is obtained by fitting an example entity set (training set), and its main usefulness is the ability to make predictions on the value for the output feature of the entity, provided the process to build the function is carried out successfully. Data-driven explanations are most valuable when the model is complex, in which case the explanation-building process is based on deriving another simpler function that approximates the original one.

Statistical. The most likely explanation is not always the best explanation [44], and users benefit most from the identification of causes to explain events, not mere associative relationships; however, these models based on probabilities or statistics are still useful and commonly used in XAI.

Simulation-based. Leverages the implementation of a system to imitate a process of interest in order to analyze the results that arise from running it on one or more input configurations. Some works [29, 57] have focused on exploring the role that these virtual simulations play in social explanations, for example by explaining real-world social phenomena. In general, because simulations play the role of a data-generating experiment, they do not directly provide an explanation; however, they do provide data to evaluate hypotheses within a theoretical framework, which ultimately provides the explanation. The field of *simulation-based inference* [22] has recently gained traction, and is a promising tool to be leveraged in XAI.

Contextual. Context typically refers to domain information about objects other than the ones explicitly participating as inputs to or outputs generated by the system, such as information about users, situations, and broader environment affecting the computation [7, 12, 25]. In [13], the author defines context as a "collection of relevant conditions and surrounding influences that make a situation unique and comprehensible". Thus, context-aware explanations often include extra information that is not contained in the current described situation but is part of the users' context.

2.3. HEIST*: An Application Framework for XAI in Socio-Technical Systems*

As mentioned, in this paper we continue the line of work where we presented the *Hybrid Explainable and Interpretable Socio-Technical Systems* application Framework (HEIST, for short), adopting it as the central architecture for tackling explainability in ISTSs. HEIST is a framework for guiding the implementation of hybrid XAI-based socio-technical systems via a combination of data-driven and KR-based models. Figure 1-(A) shows an overview of the architecture proposed in [50] (in turn, this model is a generalization of the one first introduced in [48]), which has six components:

- *Data Ingestion*: A variety of data sources feeds this module that is responsible for handling different issues that arise from integration of heterogeneous sources, such as cleaning, schema matching, inconsistency, incompleteness, as well as other higher-level issues such as trust and uncertainty management.
- *Sub-symbolic Services*: There are many tasks that address a variety of problems that are better handled by data-driven services; this module aids in the isolation of application scenarios for each service in order to foster faster deployment, replace with alternative implementations, and build more reliable explanations.

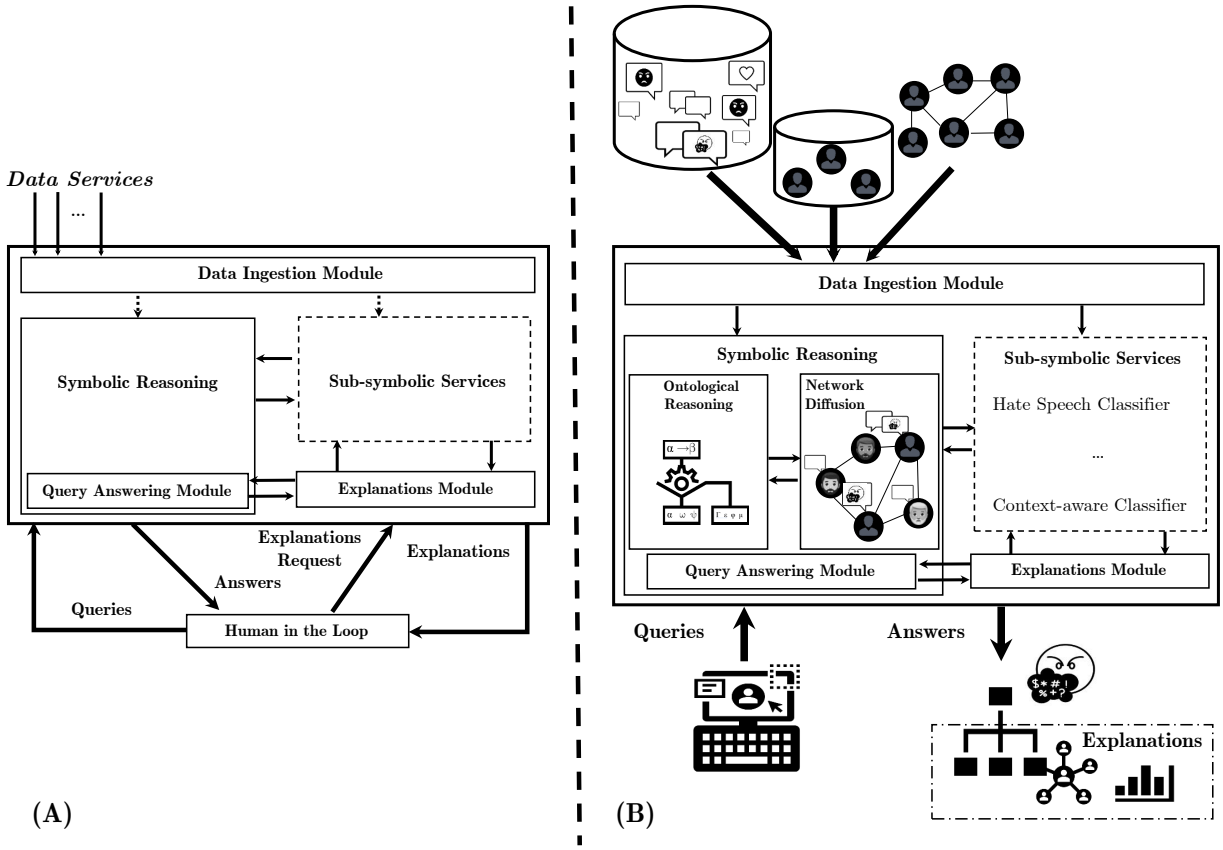

Fig. 1. The HEIST (*Hybrid Explainable and Interpretable Socio-Technical systems*) application framework (A), and its instantiation into the NETDER-HS system for detection of online hate speech (B)

- *Symbolic Reasoning*: General problems are, in many cases, better handled by high-level reasoning; this module, which can be considered the main one in the framework, is fed from both data pre-processed by the *Data Ingestion Module* and outputs generated by *Sub-symbolic Services*. In order to take advantage of inference processes, rule-based systems are typically used to carry out high-level tasks such as combining data and low-level knowledge or answering queries using well-defined reasoning mechanisms over highly-structured knowledge. The reasoning processes implemented here are the main drivers behind answering queries issued by users.
- *Explanations*: As discussed above, different kinds of explanations associated with query answers are implemented in this module, via access to the *Symbolic Reasoning* (in turn, via the Query Answering module) and *Sub-symbolic Services* modules.
- *Human in the Loop*[2]: In Social-Technical settings, the system is considered to have failed when it does not adequately consider its users' demands. This module seeks to mitigate this problem and improve system performance with iteratively administered feedback provided by human users with queries, answers, explanation requests, explanation scoring, and ranking of data sources by usefulness, among other options.
- *Query Answering*: Finally, this module is responsible for answering user queries, which requires coordinating the execution of the rest of the modules.

[2]Human-in-the-loop (or HITL, for short) is defined as a sytem/model that requires different degrees of human interaction.

Having introduced the HEIST framework, in the next section we show how a particular instantiation can be derived to produce a system for a specific use case.

3. Instantiating the HEIST Framework: The NetDER-HS System

In this section, we sketch how the HEIST framework can be instantiated for the implementation of explainable ISTSs. We begin with a brief introduction to a previously-proposed framework called NETDER, which was developed as a tool for implementing ISTSs based on the concept of automated hypothesis generation; HEIST is a generalization of NETDER where the symbolic and sub-symbolic modules are architecturally distinguished. Then, we show how the NETDER-HS system for detecting hate speech in online platforms is derived.

3.1. The NETDER *Model*

Figure 1 shows how the HEIST architecture generalizes NETDER as proposed in [48]; note that NETDER does not make explicit reference to sub-symbolic services as we do here. HEIST's Symbolic Reasoning module is instantiated in NETDER by choosing a particular knowledge representation language;. Furthermore, part of the knowledge base is assumed to encode a network of users in which diffusion processes of interest take place. This leads to two sub-modules:

- *Ontological Reasoning*: Stores the knowledge database both for background knowledge (ontological data, rules, and constraints) as well as for network knowledge (nodes, edges, and their attributes). The general HEIST module in this case is instantiated with the language chosen of existential rules (also commonly referred to as Datalog+/–) [17][3]. For the purposes of carrying out reasoning tasks, the ontological reasoning sub-module can pose queries to the *Network Diffusion* sub-module, which offers a supporting role.
- *Network Diffusion*: Responsible for modeling dynamic aspects of the network in the form of diffusion processes and performing condition checks over the network state. There are many ways in which this module can be implemented; here we will assume that a rule-based language is used for such processes, which answer queries over the network required by the Ontological Reasoning module.

Formally, NETDER knowledge bases in the Symbolic Reasoning module are defined over a relational schema $\mathcal{R}$ of the form $KB = (D, G, \Sigma, P)$ such that D is a set of atoms, G is a set of facts to allow us to define the initial network knowledge, Σ is a set of extended existential rules incorporating network knowledge, and P is a set of global and local diffusion rules where the former provide a summarized view of the global network state, while the latter are used to guide the evolution of the network state that will be necessary in the forecasts. The main reasoning task is carried out by a query answering mechanism; NETDER queries are of the form:

$$Q(\mathbf{X}) = \exists Y \rho(X_1, Y_1) \ \wedge \ \phi(X_2, Y_2) \ \wedge \ \gamma(X_3, Y_3) : [t_1, t_2]$$

where $\rho(X_1, Y_1)$ is a conjunction associated with ontological knowledge, $\phi(X_2, Y_2)$ is a conjunction associated with local network knowledge, $\gamma(X_3, Y_3)$ is a conjunction related to global network knowledge, and $[t_1, t_2]$ is a time interval where these conditions should hold. Finally, each entity in the relational schema may have associated one or more *sub-symbolic services*, which may be invoked in order to estimate the value of a property of interest of that entity.

This combination of data-driven and KR-based models is very powerful for achieving explainability, since it allows to understand the logic behind the outputs that the query answering component can provide. There are many ways in which explanations can be derived; here, we consider two main approaches, which can also be combined. First, as mentioned above, having access to the ontological knowledge is often a useful starting point in understanding the system's outputs. Second, it may be necessary to access dynamic aspects of the network diffusion

[3]Here, we choose Datalog+/– (a general family of ontology languages that is closely related to other formalisms such as Description Logics) in order to continue with the proposal in our previous work. Note that other ontology languages, such as OWL, could also be used.

process, such as the network state. In Section 4, we go into further detail by proposing several types of explanations that can be generated based on these components, particularly taking advantage of combining both knowledge- and data-driven explanation techniques.

3.2. A System for Detecting Hate Speech in Social Platforms

In this section, we instantiate HEIST/NETDER to derive a system for detecting hate speech in social platforms (an overview of this instantiation is shown in Figure 1-B); we call the resulting system NETDER-HS. We describe a simple use case where we have a typical social platform with a set of users that share information by posting and reposting content, or replying to posts from other users. In particular, we define a set of rules in the Ontological Reasoning module to work in a hate speech detection setting. Towards this end, we need to provide the details of the main components; as a first step, we assume that the *Data Ingestion* module feeds data into the ABox (ontological database) as well as the encoding of the network based on data about:

- *Posts*: Any content posted on social media platforms.
- *User profiles*: Information about users in a social platform (such as city, age, interests, etc.).
- *User relations*: Information about the existing connections between different user profiles.

We assume that the *Data Ingestion* module feeds the database available in the *Symbolic Reasoning* module with data in the following schema:

- *node*(U): U is a node of the graph, which in this case represents a user of the social platform.
- *edge*(U_1, U_2): Nodes U_1 and U_2 are connected, representing relations such as follow, friendship, fan, etc.
- *post*(P): content P has been posted.
- *posted*(U, P): User U posted P.
- *hs_level*(P, L): Post P is considered to contain hate speech by some sub-symbolic service, with associated certainty level $0 \leqslant L \leqslant 1$.
- *early_poster*(UID, P): *UID* is among the first users to share post P.
- *hyp_hatespeech*(P): Hypothesis that post P contains hate speech.
- *hyp_is_resp*(UID, P): Hypothesis that user *UID* is responsible (perhaps unwittingly) for sharing post P.
- *hyp_hater*(UID): Hypothesis that user *UID* is a "hater" (we use this term to refer to someone who knowingly posts hateful comments).
- *repeatedly_banned*(UID): User *UID* has been banned from the platform multiple times.
- *banned*(UID, T): User *UID* has been banned T times due to hate speech.
- *avg_banned*(T): T is the average number of bans per user on the platform.
- *popular*: A node is considered to be a popular user (for instance, because their posts receive a lot of attention, and they have many followers).
- *repost*(P): Post P is a reposted message.
- *offensive_language*(P, L): Post P is believed to contain offensive language with certainty level $0 \leqslant L \leqslant 1$ by some sub-symbolic service.
- *targeted*(P): Post P is targeted to an individual or group of people.
- *hs_context_level*(P, L): Post P is considered to contain hate speech, by a sub-symbolic service taking contextual information into account, with certainty level $0 \leqslant L \leqslant 1$.

Note that *hs_level*$/2$, *hs_context_level*$/2$, and *offensive_language*$/2$ are atoms derived from *external data-driven classifiers* (such as the tools developed by [59] and [28]). Hence, these can be seen as *wrappers* for a call to a sub-symbolic service. This easily allows to tune different thresholds required in the rules to be instantiated, replace one classifier with another, combine different classifiers (ensembles), and use machine learning-based predictors to solve specific tasks.

The *Ontological Reasoning* sub-module contains the logical rules shown in Figure 2 (underlined predicates correspond to wrappers for sub-symbolic services, as discussed above); we provide intuitive explanations for each rule next. Roughly speaking, rule r_1 states that given a post that the classifier detects as offensive with confidence level L (greater than the threshold 0.5) and is targeted to a person or group of people, a hypothesis is created that P is hate

Ontological Rules:

$r_1 : post(P) \wedge \underline{offensive\_language}(P, L) \wedge (L > 0.5) \wedge targeted(P) \rightarrow hyp\_hatespeech(P, red)$

$r_2 : post(P) \wedge \underline{offensive\_language}(P, L) \wedge (L > 0.2) \rightarrow hyp\_hatespeech(P, orange)$

$r_3 : post(P) \wedge \underline{hs\_level}(P, L) \wedge (L > 0.5) \rightarrow hyp\_hatespeech(P, red)$

$r_4 : post(P) \wedge \underline{hs\_level}(P, L_1) \wedge (L_1 > 0.2) \wedge (L_1 < 0.5) \wedge \underline{hs\_context\_level}(P, L_2) \wedge (L_2 > 0.5) \rightarrow hyp\_hatespeech(P, orange)$

$r_5 : post(P) \wedge posted(UID, P) \wedge hyp\_hater(UID) \rightarrow hyp\_hatespeech(P, yellow)$

$r_6 : hyp\_is\_resp(UID, P_1) \wedge hyp\_is\_resp(UID, P_2) \wedge (P_1 \neq P_2) \rightarrow hyp\_hater(UID)$

$r_7 : hyp\_hatespeech(P, L) \wedge early\_poster(UID, P) \rightarrow hyp\_is\_resp(UID, P) : \langle viral(P), [0.7, 1] \rangle$

$r_8 : hyp\_is\_resp(UID, P_1) \wedge repeatedly\_banned(UID) \rightarrow hyp\_hater(UID)$

$r_9 : hyp\_is\_resp(UID, P_1) \wedge banned(UID, T_2) \wedge avg\_banned(T_2) \wedge (T_1 > T_2) \rightarrow repeatedly\_banned(UID)$

Diffusion Rules:

$loc\_rule_1 : repost(P)_{\langle popular, [0,0] \rangle} \overset{1}{\leftarrow} (\mathsf{T}, \langle repost(P), [1, 1] \rangle \wedge \langle popular, [1, 1] \rangle,\ if)$

$glob\_rule_2 : viral(P) \leftarrow repost(P)_{\mathsf{T}}, avg$

Fig. 2. Rules in the *Ontological Reasoning Module*. Underlined predicates correspond to wrappers for an underlying sub-symbolic service.

speech with high danger level (*red*); on the other hand, r_2 expresses that a hypothesis that P is hate speech with medium danger level (*orange*) is created if P is an offensive post (that is, the targeting condition is removed) with a lower threshold for its confidence level. Rule r_3 generates a hypothesis that P is hate speech with danger level *red* if P is detected as hate speech with confidence $L > 0.5$; rule r_4 is a variant that uses a context-based classifier when $L > 0.2$ and $L < 0.5$. If the user who posted P is suspected to be a hater, then rule r_5 generates hypotheses for the existence of hate speech with lower danger level (*yellow*). Observe that the sixth and eighth rules produce hypotheses stating the existence of haters; being the user *UID* responsible for disseminating two hate speech posts is enough for generating the hypothesis that *UID* is a hater, whereas having been repeatedly banned is also a reason for generating the hypothesis that *UID* is a hater. Finally, the seventh and ninth rules can be read as: "If post P is suspected to contain hate speech (with any danger level), and user *UID* is considered to be an early poster of P that is flagged as viral on the network (with confidence at least 0.7), then there exists a hypothesis that *UID* is responsible for disseminating hate speech", and "the hypothesis that *UID* is repeatedly banned is created if the number of bans for *UID*'s account is above the average number of bans per user".

For the *Network Diffusion* sub-module, we could implement a set of logical diffusion rules with the purpose of guiding the evolution of (uncertain) knowledge about the network; an example of a language for expressing such rules is MANCaLog [60]. For this particular example, Figure 2 shows two diffusion rules, loc_rule_1 and $glob\_rule_1$, where the former is a local diffusion rule that intuitively states "users that are not popular with certainty and have popular neighbors who reposted P with certainty, will update the belief that P should be reposted, according to influence function *if*". Here, we assume that function *if* always returns the interval $[1, 1]$, meaning that P is likely to be reposted by the node in the next time point. Likewise, $glob\_rule_1$ is a global diffusion rule that allows us to flag a post as viral based on the forecasting of reposts for each node. Note that this rule involves a function *avg* that will be used to forecast the viral state of each post and it returns the required value by computing an average for the values associated with each of the node's reposts. We refer the reader to [48] and [49] for further discussions about diffusion rules.

Both the ontological KB and network database can be useful for the construction of complex explanations for the outputs generated by NETDER-based systems. In the next section, we will study several explanation types, with a special focus on explainability in NETDER. The NETDER-HS system will be adopted as the use case to show concrete examples; in Section 4.4, we extend this use case to cyberbulllying scenarios.

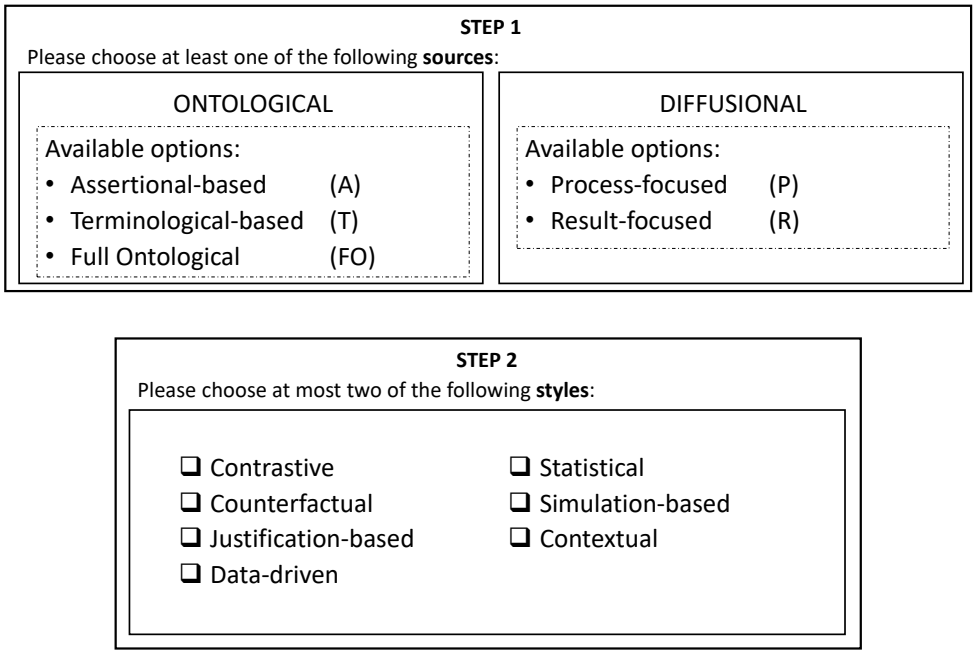

Fig. 3. Wireframe of the interface for the proposed explanation customization process.

4. Towards Concrete Explanations in the Context of HEIST

We now focus on producing concrete types of explanations in the context of systems implemented using the HEIST application framework; in order to enable a detailed discussion, we assume the pre-instantiation of HEIST into NETDER as discussed above. We begin by proposing a process by which a user can choose and combine explanation styles as required; this is done by selecting among five different sources of explanatory knowledge introduced in Section 4.1, and then requesting the specific style(s) as discussed in Section 4.2.

Build Your Own Explanation

Figure 3 illustrates in a wireframe layout a general graphical representation of a process for choosing custom-built explanations. This process can be outlined as a two-step sequence:

Step 1: Select the information sources to be used in building the explanation. This choice may include a single source or all of them. This is a first filter over the explanation styles that will be available in Step 2.

Step 2: Select the style(s) required for delivering the explanation. As not all combinations are compatible, the system can simply gray out the ones that cannot be chosen as the user makes selections.

So, as an example, we might be interested in a counterfactual explanation that is built based solely on terminological knowledge. In the next section, we go into greater depth about the different sources and explanatory styles shown in Figure 3, developing the *single-style* options into more detail. Later, in Section 4.3, we will analyze the combination of pairs of explanatory styles.

4.1. Sources of Explanatory Knowledge

We now study the range of possibilities for leveraging the different components of an ISTS based on the HEIST application framework. We consider four possible knowledge sources and their associated explanations: Terminological-based (T), Assertional-based (A), Process-focused (P), and Result-focused (R); a fifth option, which combines the first two, is the Full Ontological (FO) source. Next, we describe them in more detail.

Terminological-based (T) explanations. Explanations may benefit from extracting information and insights contained in the ontological rules/axioms; leveraging this source allows us to understand why the system provides an answer for a specific query by highlighting the rules involved in its derivation. For example, in a social platform setting like the one in the previous section, pointing to the specific rules used in reaching a conclusion that a user is suspected to be a "hater" may be a useful way to convey to users why this inference was made.

Assertional-based (A) explanations. Atomic formulas (or assertional axioms) are basic objects that in our context refer to elements present in both the ontological and network databases. Choosing this source thus allows to incorporate elements from this basic set which, as we will see, is often connected to several explanation styles arising in data-driven AI techniques. For example, an explanation may use the data available on a social platform user to show why they are classified as potentially malicious.

Full Ontological-based (FO) explanations. These explanations are fed from both atomic formulas and ontological rules. In this context, this source allows leveraging the application of a greater number of explanation styles since it is possible to exploit more information to be presented to users. For example, a trace-based explanation providing information (in form of rule trace) about inference process or network knowledge-based explanations may benefit from this type of knowledge source.

Process-focused (P) explanations. In this case, explanations are designed with the aim of shedding light on how the network model evolves as the execution of the diffusion process progresses from the initial state to the final state (when the model is ready to make a prediction). For example, if a user posts a comment containing offensive language, a rule may be fired incrementing the probability that the user is a hater (as in the use case above), producing an evolution of the network model modifying the initial state. Then, having a user with this probability above a certain threshold may fire another rule issuing a prediction that the user will likely post such content again in the future, and should be flagged with a first warning. Explanations of this kind would then highlight the rules used to update the network model during the process; in general, if the diffusion process is not rule-based, explanations can highlight the transitions that the model goes through.

Result-focused (R) explanations. These explanations focus on the final result of the diffusion process over the network, which can be seen as a forecast of the network's evolution over time. For example, an explanation as to why a post has been flagged may highlight how the post will "go viral" by showing the number of users it is expected to reach in the next three days.

In the next two sections, we describe in detail how single-style and combined-style explanations arise in the context of the seven basic styles[4] and five sources discussed above. In each case, we discuss the results of an analytical evaluation carried out to study the feasibility of each primitive style-source pair (cf. Figure 4 for the results) as well as the added complication of deciding in which order to apply styles when two are selected (results shown in Figure 5).

4.2. Single-style (Primitive-only) Explanations

We now present additional details regarding the primitive explanation styles discussed in Section 2.2, providing initial details on each primitive style and examples in each case depending on the sources that are available, as presented in Figure 4.

Contrastive explanation. Consider the following elements involved in this explanation style:

- A NETDER knowledge base $KB = (D, G, \Sigma, P)$.
- A NETDER query $CQ(\mathbf{X})$ where $\mathbf{X} = X_1, \ldots, X_n$.

[4]The set of basic styles that we work with in this paper is not exhaustive; the functionality of future versions of HEIST may be incremented by incorporating other styles from the XAI literature.

Primitive Explanation Styles	Sources				
	Assertional (**A**)	Terminological (**T**)	Full Ontological (**FO**)	Process-focused (**P**)	Result-focused (**R**)
1. Contrastive	✓	✓	✓	✓	✓
2. Counterfactual	✓	✓	✓	✓	✓
3. Justification-based	✓	✓	✓	**?**	**?**
4. Data-driven	✓	×	×	×	×
5. Statistical	✓	✓	✓	✓	✓
6. Simulation-based	×	×	×	✓	✓
7. Contextual	✓	✓	✓	✓	✓

Fig. 4. Sources available to each primitive explanation style. Checkmarks indicate availability, crosses indicate non-availability, and question marks are used when sources are potentially available (exploring the specific conditions is outside the scope of this work).

– Two Boolean NETDER queries BCQ and BCQ' of the form $BCQ = CQ(X_1 = v_1, \ldots, X_i = v_i, \ldots, X_n = v_n)$ and $BCQ' = CQ(X_1 = v'_1, \ldots, X_i = v'_i, \ldots, X_n = v'_n)$, where it holds that $v'_i \neq v_i$ for some $1 \leqslant i \leqslant n$ and $similar(BCQ, BCQ')$.

A contrastive explanation for an answer to BCQ over KB is the pair $\langle (BCQ, a), (BCQ', a') \rangle$, where $a, a' \in \{Yes, No\}$ and are the answers to BCQ and BCQ', respectively.

Note that function $similar(Q_1, Q_2)$ requires implementation in order to yield a similarity value given two queries (cf. Class `ContrastiveStyle` in Section 5).

Example: Consider the application domain introduced in Section 3.1, and the user query $hyp\_hater(john)$, which has answer *Yes*. Assuming that $similar(hyp\_hater(john), hyp\_hater(paul))$ holds, the contrastive explanation is: $\langle (hyp\_hater(john), Yes), (hyp\_hater(paul), Yes)) \rangle$, which can intuitively be interpreted as "user *John* is a hater because user *Paul* is similar to *John*, and he is a hater, too".

Counterfactual explanation. This style yields explanations built using knowledge that is *not included in any knowledge source*. Concretely, consider a Boolean query Q, and a NETDER knowledge base $KB = (D, G, \Sigma, P)$. A counterfactual explanation for an answer to Q over KB is based on both available and not available knowledge; that is, it produces $KB' = (D', G', \Sigma', P')$ by changing elements in KB (such as data and/or rules) to be used to answer Q.

Example: Continuing with the example illustrated in Figure 2, let $hyp\_hatespeech(post_1, yellow)$ be a query, which we assume has answer "No". A counterfactual explanation to this answer can be generated by adding new information to the knowledge base, such as the fact that a user who posted $post_1$ is identified as a hater. Furthermore, suppose that with this new information rule r_5 can now be applied and the answer for our query now becomes "Yes"[5]. A counterfactual explanation in this case could then be "there would be a hypothesis for $hyp\_hatespeech(post_1, yellow)$ if it were the case that the user who posted $post_1$ is suspected to be a hater".

Justification-based explanations. Suppose we have a Boolean NETDER query Q, a NETDER knowledge base $KB = (D, G, \Sigma, P)$, and that the answer for Q over KB is "Yes". Then, a justification-based explanation for this answer is another NETDER knowledge base $KB' = (D', G', \Sigma', P')$ such that:

– $D' \subseteq D$,
– $G' \subseteq G$, $\Sigma' \subseteq \Sigma$,
– $P' \subseteq P$,
– the answer for Q over KB' is also "Yes", and
– for all $KB'' = (D'', G'', \Sigma'', P'')$ if $D'' \subset D', G'' \subset G', \Sigma'' \subset \Sigma'$, and $P'' \subset P'$, then the answer for Q over KB'' is "No". That is, it must include only the minimal necessary knowledge.

[5]Though it is not strictly necessary for the answers over KB and KB' to be different, it is often useful for this to be the case.

Justification-based explanations only make sense when the answer is 'Yes', since otherwise there does not exist a minimal subset of knowledge to support the query. Figure 4 indicates that this kind of explanation is available with sources from the ontology (A, T, and FO); in the case of P and R-sourced explanations, the same criteria could be applied, though this will depend on the specific language used to implement the diffusion process. Exploring this issue further is the topic of future work.

Example: Consider the ontological rules presented in Figure 2, and let $hyp\_hatespeech(post_1, red)$ be a Boolean query. Suppose we can create a hypothesis for $hyp\_hatespeech(post_1, red)$, which can be obtained by applying rule r_1. From our definition of justification-based explanation, we can generate an explanation with r_1 and the set of atoms $\{post_1, offensive\_language(post_1, 0.6), targeted(post_1)\}$, which we assume holds. From the point of view of an ontology engineer, this type of explanation would for instance help carry out debugging tasks by pinpointing what causes $hyp\_hatespeech(post_1, red)$ to hold.

Data-driven explanations. In this context, we have the following central elements:

- A NETDER knowledge base $KB = (D, G, \Sigma, P)$ over the relational schema $\mathcal{R}$.
- A set of entities $\mathcal{E} \subseteq \mathcal{R}$.
- An instance $e(v_1, \ldots, v_n, v_{n+1})$ of an entity $e \in \mathcal{E}$.
- A set of sub-symbolic services $\mathcal{S}$, and a sub-symbolic service $s_1 \in \mathcal{S}$ associated with entity e.
- A set E of instances $e(v'_1, \ldots, v'_n, v'_{n+1})$ of entity e.
- A threshold $\theta_1 \in [0, 1]$, and a comparison operator $op \in \{>, <, =, \neq, \geqslant, \leqslant\}$.
- A value $v_{n+1} \in [0, 1]$ that results from invoking s_1 over $(v_1, \ldots, v_n)$.

Then, we say that a data-driven explanation of the answer for query $e(v_1, \ldots, v_n, v_{n+1}) \wedge (v_{n+1}\ op\ \theta_1)$ over *KB* is an explanation based on a function f such that for all $e(v'_1, \ldots, v'_n, v'_{n+1}) \in E$ we have $f(v'_1, \ldots, v'_n) = v'_{n+1}$ and $v'_{n+1} \in [0, 1]$ is also the result of invoking s_1 over $(v'_1, \ldots, v'_n)$.

As shown in Figure 4, this primitive style is only available for the assertional source.

Example: We can assume that a data-driven explanation request is posed in order to know the reasons behind the answer "Yes" for the query $offensive\_language(post, level) \wedge (level > 0.5)$, where *post* is a value based on textual content using offensive language with confidence value *level* predicted by the sub-symbolic service s_1. Hence, a data-driven explanation could rely on a function f obtained from variants of the entity *offensive_language* and the result of predictions for these variants made by s_1; in this case, it can be used in the provided explanation in order to unveil the most important words involved in the assertion "*offensive_language* is used in *post*".

Statistical explanation. This kind of explanation is likely the most general and flexible of the primitive styles we consider, since it allows to summarize in a numeric way the reasons involved in obtaining a specific output, making it possible to apply them in a variety of situations. Consider a Boolean NETDER query Q and a NETDER knowledge base $KB = (D, G, \Sigma, P)$; we say that a *statistical explanation* of an answer to Q over *KB* is an explanation based on numeric indicators from functions that summarize the occurrence of data related to the answers of Q over *KB*. Examples of such functions required for these explanations are the well-known aggregation functions such as max, min, classical statistical functions such as mean, median, standard deviation, as well as more complex ones.

Example: Consider again the query $hyp\_hater(paul)$, and assume that the answer is "Yes". Examples of these explanations can include outputs of functions used to calculate the number of occurrence of posts, bans, comments, and specific connections between users such as follow, friendship, likes, etc. More concretely, network atoms associated with the current network state (encoded by atoms as *node*, *edge*, *posted*, among others) offer interesting alternatives when computing such statistics.

Simulation-based explanation. Assume a Boolean NETDER query Q and a knowledge base $KB = (D, G, \Sigma, P)$; a *simulation-based explanation* of an answer to Q over *KB* is built based on imitating the evolution over time of a network diffusion process that provides insight as to why it is possible to answer Q using *KB*. As indicated in Figure 4, this primitive style is only available with the process- and result-focused sources.

Example: An explanation based on a forecasting model about the evolution of content in the network may explain why a post has been flagged as viral. Consider the diffusion rule $diff\_rule_1$ presented in Figure 2; a possible expla-

nation may be to show how a diffusion process models the future spread of hate content after three days, indicating estimates of the percentage of users that will repost such content.

Contextual explanation. Consider a Boolean NETDER query Q and a NETDER knowledge base $KB = (D, G, \Sigma, P)$; a *contextual explanation* of an answer to Q over KB is built including knowledge that does not participate in the computation. The explanation may thus contain knowledge that, while not used to answer Q over KB, is otherwise related to the set of answers.

Example: Considering the previous explanation scenario, the information about users that reposted hate content could serve as a contextual explanation. This may include additional information about them (such as location, date of last post, etc.), and the users' closest network relations.

4.3. Combining Explanations

In requesting an explanation, our proposal provides facilities for generating explanations that could combine several styles and knowledge sources. In order to better understand these combinations, we carry out here a preliminary analytical study about which explanations can be built using the styles and sources addressed in this paper. Our main findings are summarized in Figure 5. Note that we do not consider all possible combinations between available sources and explanation styles; in these first steps towards understanding combined explanations, we analyze how to combine two single-style explanations at a time.

Different approaches could be developed for combining explanation styles; this is in general not a simple endeavor due to several characteristics, such as contemplating the combined explanation's goal, stakeholders, application domain, among others. For the purposes of this work, we identify three basic options:

Policy 1: Obtain each of the selected styles independently and then deliver them together.
Policy 2: Generate a systematic composition of explanations by sequentially applying styles, starting from the initial source(s) of explanatory information and using the output of one as the input to the next.
Policy 3: Implement an ad hoc combination when explanations cannot be addressed independently.

Policy 1 can be applied by taking the information from Figure 4 and checking if all the chosen styles are compatible with the selected sources; this is the policy we include in our initial implementation (cf. Section 5). We now provide a preliminary analysis of the feasibility of applying Policy 2 with two styles; Policy 3 requires an analysis at a lower level of abstraction, and will be addressed in future work.

Combining Two Different Styles

In order to apply Policy 2, we need to understand what combinations are useful and comprehensible to make sense of the resulting explanation. In the following, we discuss the central aspects that emerge from considering the pairwise combinations shown in Figure 5; in some cases, the scope of the discussion may be limited by the fact that concrete implementation details of each particular style, as well as the explanation generation process, are domain-specific. Our discussion will be organized according to the blocks in which the table in Figure 5 is separated.

We begin with an analysis of three groups of combinations that share an important feature that causes order to be important; in particular, in each case there is a primitive style that generates elements to be used in the explanation that can later naturally be used as inputs to other styles:

- *Counterfactual + another primitive style (Styles 8–13):* The counterfactual style involves deriving another knowledge base, which can then be taken as input for the other style.
- *Contrastive + another primitive style (Styles 14–18):* The contrastive style provides a pair of queries and their answers, which can in turn be used by the second style.
- *Justification-based + another primitive style (Styles 19–22):* The justification-based style identifies a subset of the knowledge base that, analogously to the combination with counterfactuals, can be taken as input by the other style.

Combined Explanation Styles	Sources				
	A	**T**	**FO**	**P**	**R**
8. Counterfactual (2) + Justification-based (3)	$2 \prec 3$	$2 \prec 3$	$2 \prec 3$	**?**	**?**
9. Counterfactual (2) + Data-driven (4)	$2 \prec 4$	$\times$	$2 \prec 4$	$\times$	$\times$
10. Counterfactual (2) + Statistical (5)	$2 \prec 5$	$2 \prec 5$	$2 \prec 5$	$2 \prec 5$	$2 \prec 5$
11. Counterfactual (2) + Simulation-based (6)	$\times$	$\times$	$\times$	$2 \prec 6$	$2 \prec 6$
12. Counterfactual (2) + Contextual (7)	$2 \prec 7$	$2 \prec 7$	$2 \prec 7$	$2 \prec 7$	$2 \prec 7$
13. Counterfactual (2) + Contrastive (1)	$2 \prec 1$	$2 \prec 1$	$2 \prec 1$	$2 \prec 1$	$2 \prec 1$
14. Contrastive (1) + Justification-based (3)	$1 \prec 3$	$1 \prec 3$	$1 \prec 3$	**?**	**?**
15. Contrastive (1) + Data-driven (4)	$1 \prec 4$	$\times$	$1 \prec 4$	$\times$	$\times$
16. Contrastive (1) + Statistical (5)	$1 \prec 5$	$1 \prec 5$	$1 \prec 5$	$1 \prec 5$	$1 \prec 5$
17. Contrastive (1) + Simulation-based (6)	$\times$	$\times$	$\times$	$1 \prec 6$	$1 \prec 6$
18. Contrastive (1) + Contextual (7)	$1 \prec 7$	$1 \prec 7$	$1 \prec 7$	$1 \prec 7$	$1 \prec 7$
19. Justification-based (3) + Statistical (5)	$3 \prec 5$	$3 \prec 5$	$3 \prec 5$	**?**	**?**
20. Justification-based (3) + Simulation-based (6)	$\times$	$\times$	$\times$	**?**	**?**
21. Justification-based (3) + Contextual (7)	$3 \prec 7$	$3 \prec 7$	$3 \prec 7$	**?**	**?**
22. Justification-based (3) + Data-driven (4)	$3 \prec 4$	$\times$	$3 \prec 4$	**?**	**?**
23. Data-driven (4) + Statistical (5)	$4 \prec 5$	$\times$	$\times$	$\times$	$\times$
24. Data-driven (4) + Simulation-based (6)	$\times$	$\times$	$\times$	$\times$	$\times$
25. Data-driven (4) + Contextual (7)	✓	$\times$	$\times$	$\times$	$\times$
26. Statistical (5) + Simulation-based (6)	$\times$	$\times$	$\times$	$6 \prec 5$	$6 \prec 5$
27. Statistical (5) + Contextual (7)	✓	✓	✓	✓	✓
28. Simulation-based (6) + Contextual (7)	$\times$	$\times$	$\times$	$6 \prec 7$	$6 \prec 7$

Fig. 5. Combined explanation styles against the different types of knowledge sources that can be used in explanations. Check marks indicate that it is possible to build the explanation applying the styles in any order, while "$\prec$" indicates that a specific order must be followed. Finally, $\times$ and ? indicate that the combination of styles and source is not possible (or not fully explored, respectively).

The table in Figure 5 shows these resulting orders and how they can be combined with the five different sources to derive combined explanations. Consider the following example for explanation style 14 (Contrastive + Justification-based) that arises from combining the examples included in Section 4.2. Upon requesting the contrastive style (cf. Page 12), the system yields $\langle(\mathit{hyp\_hater}(\mathit{john}), \mathit{Yes}), (\mathit{hyp\_hater}(\mathit{paul}), \mathit{Yes})\rangle$. If a justification-based explanation is then requested over this output, one possible explanation would be that $\mathit{hyp\_hater}(\mathit{john})$ was derived using rules r_7 and r_8, while $\mathit{hyp\_hater}(\mathit{paul})$ was derived via r_6 and r_8 (and the atoms that caused each rule to fire).

Styles 23–27: In this group, we have four possibilities. In the first two, we have Data-driven with either Statistical or Contextual styles; for the former, the most natural order is to first apply the Data-driven style, which can then be fed to the Statistical style, while for the latter, both orders could in principle be applied: Data-driven $\prec$ Contextual, or the other way around. Similarly, for Style 26, the Statistical style is most naturally applied last, while in the case of Style 27 both orders are possible. For an illustrative example of style 23 (Data-driven + Statistical), consider the example from Section 4.2 (cf. Page 13) where the user requests a data-driven explanation for $\mathit{offensive\_language}(\mathit{post}, \mathit{level}) \wedge (\mathit{level} > 0.5)$ and is shown the set of most relevant words involved in that assessment. If an additional Statistical explanation is requested, the system may provide the information regarding the percentage of times in which posts containing those words were correctly classified as offensive.

Finally, in Style 28 the most sensible order is Simulation-based $\prec$ Contextual, given that a simulation may be enhanced by contextual information.

Having presented the set of primitive and combined explanation styles, we now come back to the running example and present another possible application in the domain of detecting unwanted behavior in social platforms.

4.4. A Cyberbullying Detection Use Case

In order to further illustrate how the different explanation styles offered in HEIST can be leveraged in a real-world setting, we now develop an additional example to show how the NETDER-HS system can be extended to be used in a simple cyberbullying detection case. More concretely, we present a set of additional rules and then discuss how several explanations can be derived so that domain experts can get a clearer understanding of outputs generated by HEIST-based systems.

As a first step, we extend our hate speech example already presented in Section 3.2 with new rules. The following rules drive the generation of hypotheses for online cyberbullying behavior:

- $r_{10} : \mathit{post}(P) \wedge \underline{\mathit{offensive\_language}}(P, L) \wedge (L > 0.5) \wedge \mathit{targeted}(P, U) \rightarrow \mathit{hyp\_bullying}(U)$

  If a post P contains offensive language with confidence $L > 0.5$ and it is targeted towards a particular individual U, then a hypothesis is generated that U is a bullying victim.

- $r_{11} : \mathit{hyp\_hatespeech}(P, L) \wedge \mathit{targeted}(P, U) \rightarrow \mathit{hyp\_bullying}(U)$

  If P is targeted towards U and suspected of containing hate speech (with any danger level), then a hypothesis is created that U is being bullied.

- $r_{12} : \mathit{posted}(UID, P) \wedge \mathit{fake\_profile(UID)} \wedge \mathit{fake\_news}(P) \wedge \mathit{targeted}(P, U) \rightarrow \mathit{hyp\_bullying}(U)$

  If UID uses a fake profile for posting fake news about U, then a hypothesis that U is being bullied is created.

Having introduced how the knowledge base in the system's *Symbolic Reasoning Module* is set up, we now focus on how explanation sources and styles (as presented in Figure 3) can be used to build different explanations for a specific query. We contemplate how a range of such different explanation requests can be addressed; as discussed in Section 4.3, we assume that the system follows Policy 2, which solves the problem of combining explanation styles by sequentially applying the chosen ones (the output of one becomes the input to the next).

The first step is to choose the information source to be used; independently of the selected styles, in this particular example we assume that users always choose to use all of the available sources in building explanations. The second step is to choose the specific explanation style(s) that will be presented. We now briefly discuss how several different explanations can be generated by combining specific pairs of styles.

Counterfactual + another primitive style

Suppose that we are interested in knowing whether user *Paul* is a victim of bullying posts, and let's assume that the answer to the corresponding query $\mathit{hyp\_bullying}(\mathit{paul})$ is "*No*". As shown in Section 4.2, building counterfactual explanations involves considering information that is not present in the knowledge base. In this case, the negative answer to our query implies that rules r_{10}–r_{12} are not applied, so possible counterfactual explanations would require satisfying at least one of their bodies; for instance, a user hiding behind a fake identity could post a fake news article with content about *Paul*, leading to rule r_{12} firing and the answer changing to *Yes*. An explanation arising in this manner could be:

> "There would be a hypothesis for $\mathit{hyp\_bullying}(\mathit{paul})$ if it were the case that there is a user suspected of having a fake profile and posting a fake news article about *Paul*".

The suspected user could be chosen among the set of known users, for instance by focusing on the ones who are closest to being suspected of using a fake account. Let's further suppose that a contextual explanation request is posed by a user who wants to know more about the anonymous user who helped fire the rule. In this case,

information about account creating date, number of targeted fake posts, and followers could serve as inputs to building a contextual explanation.

Finally, we may wish to obtain a contrastive explanation in connection with our counterfactual explanation. In this case,

$$\Big\langle (\mathit{fake\_profile}(\mathit{mary}), \mathit{Yes}), (\mathit{fake\_profile}(\mathit{john}), \mathit{Yes})) \Big\rangle$$

could be considered a contrastive explanation that is read as follows: "User *Mary* often uses fake profiles because user *Paul* is similar to *Mary*, and he uses fake profiles, too".

Contrastive + another primitive style
Suppose the user selects combined explanation style 9 (Contrastive + Statistical). Considering the contrastive explanation presented above, statistical explanations for $\mathit{fake\_profile}(\mathit{mary})$ can include for instance outputs of functions based on the number of posts by *Mary* and how many followers she has. If a simulation-based explanation is required, it may be generated via the use of the network diffusion simulation model to forecast who will participate in the diffusion of *Paul* and *Mary*'s posts for the next 5 days.

Justification-based + Data-driven
Suppose a hypothesis $\mathit{hyp\_bullying}(\mathit{mary})$ is derived by applying rule r_{10}, and so the answer to our query is "*Yes*". In the case that a justification-based explanation built with solely atomic formulas was required, the system can build an explanation with the atoms[6]

$$\Big\{\mathit{post}(\mathit{post}_1), \quad \underline{\mathit{offensive\_language}}(\mathit{post}_1, 0.6), \quad \mathit{targeted}(\mathit{post}_1, \mathit{mary})\Big\}.$$

Upon demanding a Data-driven explanation for $\underline{\mathit{offensive\_language}(\mathit{post}_1, 0.6)}$, the system could apply the well-known LIME [53] approach and show the importance given to the most relevant words (based on relevance scores) that contributed to the prediction.

Data-driven, Statistical, Simulation-based, and Contextual
Consider again the data-driven explanation presented above; if explanation style 25 (Data-Driven + Contextual) is required, the system may provide information related to post post_1 as a network knowledge-based contextual explanation, such as number of reposts, number of humiliating words, and relationship between users reposting post_1. In the case of contextual explanations derived from a statistical explanation (style 27), an interesting example may include information regarding the users that arise from analyzing the percentage of anonymous users reposting hostile messages (statistical explanation). Finally, suppose we create a simulation-based explanation that is built leveraging information regarding a diffusion forecasting model of *Paul* and *Mary*'s posts (cf. the example of *Contrastive + Simulation-based* presented above); then, the system may show the number of suspected anonymous users as contextual explanatory information.

Statistical + Simulation-based
As discussed above, the most natural order when combining these primitive styles involves first obtaining a simulation-based explanation. Suppose we generate a hypothesis for $\mathit{hyp\_bullying}(\mathit{mary})$, which is derived by applying rule r_{11}. In this case, we assume that there exists a post targeting *Mary* that contains hateful comments. Suppose we create the simulation-based explanation illustrated in Section 4.2; then, the percentage of suspected anonymous users that have reposted hate speech posts may serve as contextual explanation.

Finally, with this use case illustrating additional examples, we have shown how our approach can be used in a different domain requiring supporting information for different query answers, which highlights the capacity of our approach to flexibly embed different types of explanations. Note that concerns regarding the design of interfaces to

[6]Recall that in our running example we use the convention that underlined atoms are those that have an underlying ML-based tool in the process of deciding their truth value.

present explanations to users are outside of scope of this work; nevertheless, it is important to note that there have been many R&D efforts towards this direction. While there are common approaches that communicate explanation structures with textual explanations or simple lists of elements, complex concepts are often better explained (and deeper insights are obtained) with richer visualization techniques, or via a combination of visualizations and simpler presentations.

5. The Main Implementation Tools

In this section, we discuss in detail the software engineering aspects involved in a preliminary implementation of our general framework, and how it can be used in practice. The content is organized as follows:

- Detailed presentation of the development of the HEIST application framework, providing publicly-available code specifying its structure.
- Specification of how HEIST can be extended to obtain the NETDER model.
- Illustration of the workflow involved in the explanation request activity, which is the main focus of this work.

The source code[7] for this initial version of the HEIST application framework is publicly available[8].

5.1. The HEIST Framework

Figure 6 shows a high-level view of the class diagram for the HEIST framework (we include the detailed version of this diagram in Appendix A); this diagram corresponds to a more detailed view of Figure 1 (A). In the following, we describe the most important links between the classes, which appear as relations in the diagram. For the purposes of this work, we assume that the symbolic reasoning module will be implemented as a rule-based system (*i.e.*, the knowledge base is comprised of a set of rules and a set of atomic formulas referred to as a database).

One of the main components in HEIST is the Data Ingestion module, which is in charge of creating the knowledge base and also communicates with the Sub-symbolic Services module in order to both train new data-driven tools and make use of others already in place. The Symbolic Reasoning module implements a series of reasoning tasks, each of which involves a specific inference mechanism that functions by leveraging specific knowledge sources. Note that the examples in this paper are based on a use case in which existential rules are used as the ontology language; however, HEIST allows the use of other languages, such as the well-known Web Ontology Language (OWL) or others by instantiating the relevant classes accordingly (cf. Figures 6 and 9). The Query Answering module is in charge of delivering the main system functionality, which depends primarily on the Symbolic Reasoning module (which, in turn, is supported by the Sub-ssymbolic Services). The main client of the Query Answering module is the User, who issues queries and may also request explanations for their answers. In that case, the Explanations module interacts with: (i) the Query Answering module, in order to obtain the available sources of knowledge; (ii) the Sub-symbolic Services module, to have access to the sub-modules that are related to the query answers; and (iii) the User, to allow them to choose the source(s) and style(s) for the requested explanation.

Finally, in designing this class structure, the Strategy design pattern was applied for the `ReasoningTask`, `Explanation`, `Source`, and `Style` classes. For the `StyleVisitorFactory` class, a combination of the AbstractFactory and Visitor patterns were applied.

5.2. The NetDER Model as an Extension of the HEIST Framework

Figure 7 shows the high-level class diagram for the instantiation of HEIST that produces the NETDER model; this diagram corresponds to a more detailed view of Figure 1 (B). Compared to the previous section's diagram, here we can see that this model includes the result of making several implementation choices:

[7] Code for the implementation of an application framework is not executable code, but rather a set of abstract classes that can be used as templates for developing concrete systems.

[8] https://github.com/jnparedes/HEIST

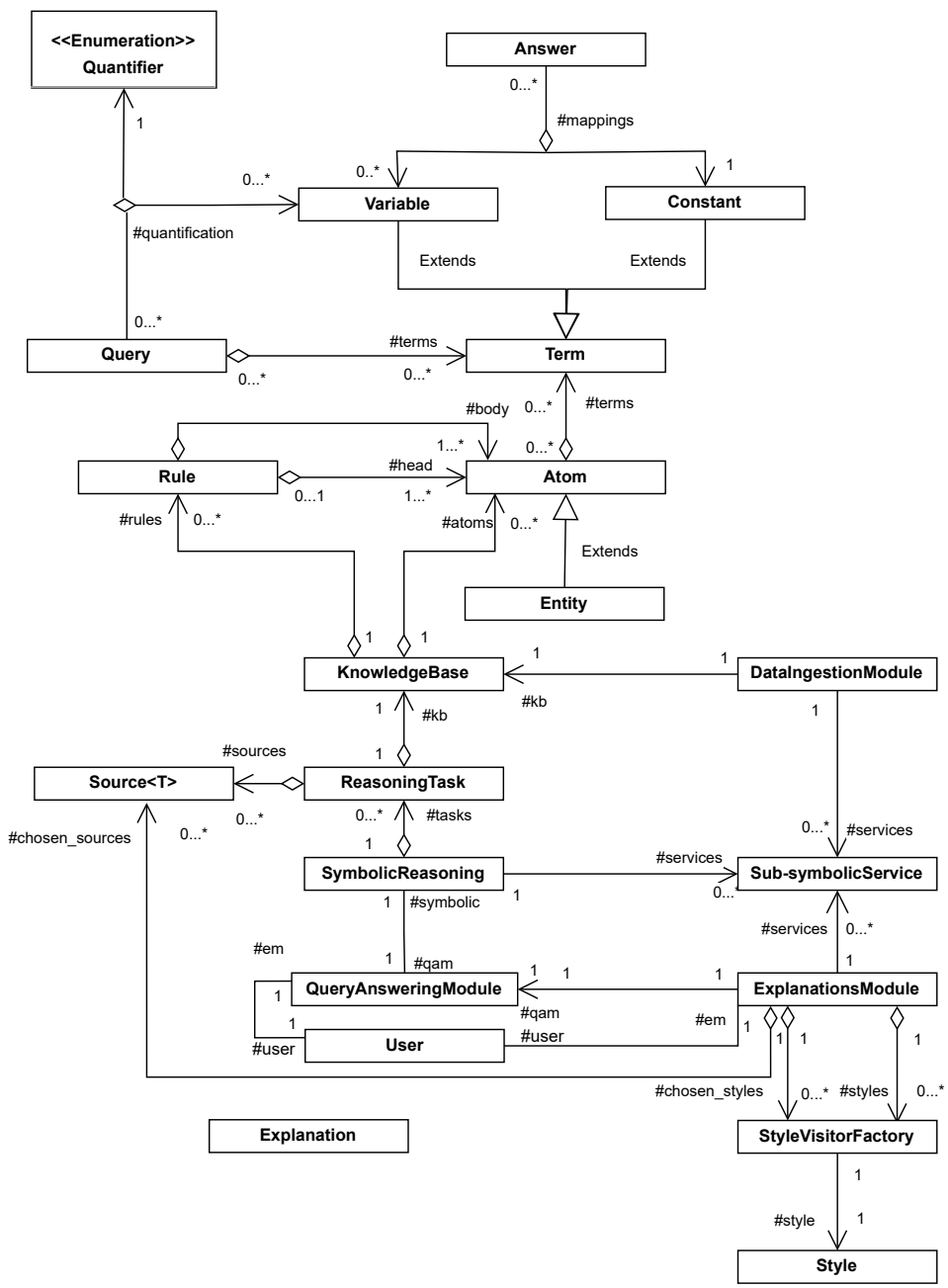

Fig. 6. High-level view of the class structure of the HEIST application framework (for detailed view, see Figure 9 in Appendix A).

– Reasoning Tasks: `Chase` (for query answering with existential rules) and `Diffusion` (which is used to represent and query diffusion processes in networked data).
– Knowledge Sources: The `Source` class is used to implement the knowledge sources, of which we have four: `Terminological` and `Assertional`, to be used by `Chase`, and `Process-focused` and `Result-focused`, to be used by `Diffusion`.

– Explanation Styles and their associated Explanations: we have one class for each primitive style discussed in Section 4.2 and corresponding explanation:

  * `ContrastiveStyle` and `Contrastive`
  * `CounterfactualStyle` and `Counterfactual`
  * `Justification-basedStyle` and `Justification-based`
  * `Data-drivenStyle` and `Data-driven`
  * `Data-drivenStyle` and `Data-driven`
  * `StatisticalStyle` and `Statistical`
  * `Simulation-basedStyle` and `Simulation-based`
  * `ContextualStyle` and `Contextual`

  In this preliminary version, we only consider the implementation issues for combining single-style explanations via Policy 1 (cf. Section 4.3); more complex combined-style explanations will be incorporated in a later version.

We now explore the mechanisms behind the request and delivery of explanations.

5.3. The Explanation Request Workflow

Figure 8 shows interaction diagrams for the main activity of interest for this paper: the explanation request issued by a user. The diagram at the top shows the main steps: a user carries out an explanation request for a query answer, which implies that a `User` object receives a message that invokes the `request(ans,q)` method, and then the sources are chosen in order to provide the necessary ingredients to build the explanations. Then, the explanation styles are chosen, which allow us to define the particular nature of the required explanation.

The diagram at the bottom of Figure 8 shows a more detailed view of what goes on in the `explain(ans, q)` method in `ExplanationsModule`: after sources and styles are chosen, an `ExplanationsModule` object receives an invocation of its `explain(ans,q)` method associated with a query and an answer. The `StyleVisitorFactory` classes relevant to the user's choices are then in charge of creating the necessary style objects; the chosen knowledge sources also affect the process by linking the relevant source objects; this essentially implements the source-style associations illustrated in Figure 4. After these steps, the `ExplanationsModule` object will request from each style object the associated explanations, which are returned to the user as a collection of `Explanation` objects.

6. Related work

In discussing the existing literature, we begin with work in the area of explainability, particularly in data-driven and knowledge-based systems. Explainability is a vast, multidisciplinary area that has especially received much attention in the last five years or so, and providing a complete survey is outside of the scope of this work. After covering XAI from the two main sides that are involved in our model (knwoledge-based and data-driven), we briefly discuss XAI in cybersecurity, which is the example application domain used throughout this work.

6.1. Knowledge-based Explanations

Knowledge-based systems were conceived as tools to support human decision-making, and generally consist of a knowledge base encoding the domain knowledge, usually modeled as a set of rules, a database, and a reasoner that makes use of these components to answer user queries. Though traditionally logic-based reasoning formalisms have been touted as more closely related to the way humans reason, and therefore more inherently explainable, different kinds of explanations are essential to the interaction between users and knowledge-based systems, as discussed in this paper. Thus, questions like *what* a system does, *how* it works, and *why* its actions are appropriate may not be immediately answerable even for logic-based approaches.

Many works, such as [11, 35, 71], focus on explaining a system's decisions from a range of several different explanations types, providing various levels of assistance. In this regard, when requiring domain information to

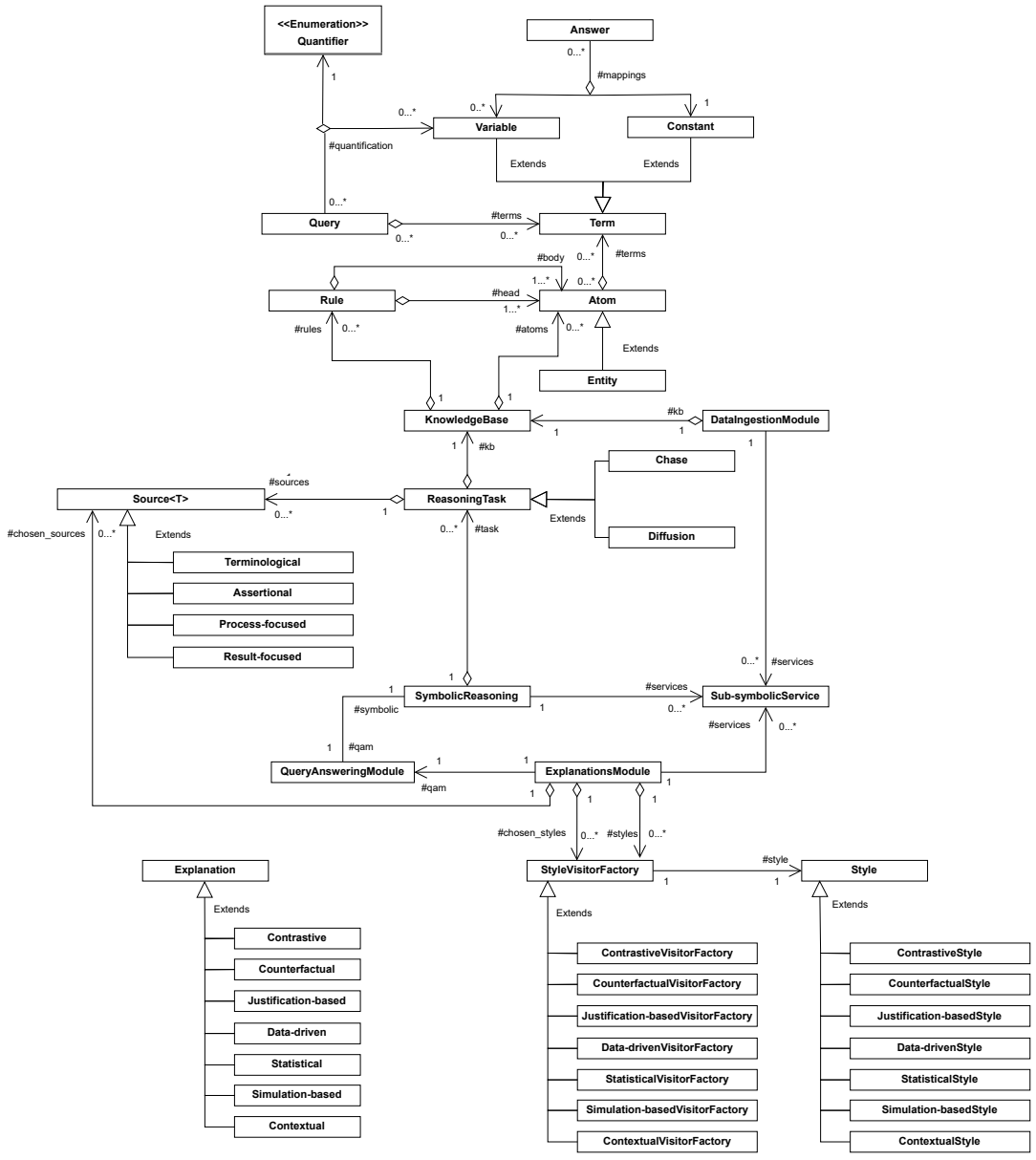

Fig. 7. High-level view of the class structure of the NETDER instantiation of HEIST.

explain a decision, structured forms of domain information like ontologies are in many cases particularly well suited. For example, there has been an important body of research directed towards the area of explanation and ontology debugging related to the Web Ontology Language (OWL); in particular, work has focused on a specific type of explanation called justifications [35, 37, 58]. Approaches like [36] have focused on implementation aspects by presenting libraries and computational tools for working with justification-based explanations of entailments in OWL ontologies. The main focus of research in this area has been on explanation for the sake of debugging problematic entailments, such as class unsatisfiability or ontology inconsistency. Other approaches regard proofs, which can contain intermediate steps between a justification and the conclusion, as a means of providing more detailed explanations [3, 9, 10]. Recent successes in machine learning technologies [21] have brought more attention

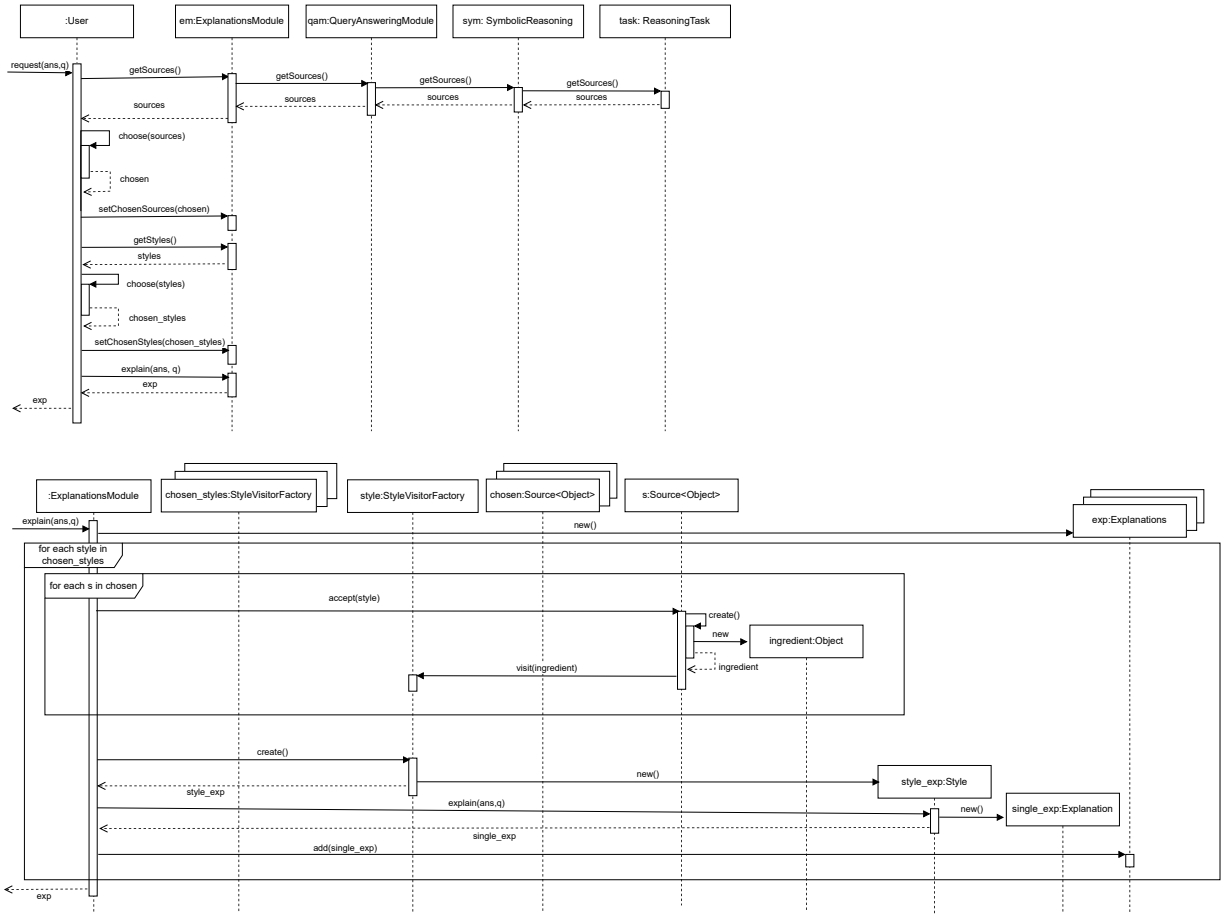

Fig. 8. Interaction diagram for the explanation request activity.

to explainability, especially on how to integrate and use domain knowledge to drive the explanation delivery process and how this influences the understanding of explanations by users. Our proposal is part of the same general effort to enhance logic-based formalisms with knowledge obtained from the use of machine learning tools. In this regard, examples of advantages of using HEIST include adaptability and interoperability with data models implemented as a service in the sub-symbolic module. Other specific works on knowledge-based explainability include approaches based on argumentation [2] and answer set programming [15].

6.2. Data-driven and Hybrid Explanations

In the last decade or so, with the availability of vast volumes of data, machine learning algorithms have seen a boom in popularity; however, these models are typically seen as difficult to understand, and there has thus been a significant focus on making them explainable. While some models are considered explainable by design (such as decision trees), others do not reveal sufficient details about how their outputs are generated, resulting in an opaque (also referred to as black box) decision model. This has led to the development of a variety of approaches for deriving explanations of opaque models, for both fully autonomous and human-in-the-loop systems. The literature on data-driven XAI is vast, and although there is no clear general consensus on what constitutes a good explanation, many taxonomies commonly used to classify explainability methods cover the following dimensions:

- *Method*: There are two main types of methods to provide explanations. Local ones give explanations via individual instances or groups of nearby instances, while global ones describe the behavior of models as a whole.
- *Stakeholder*: There are three large groups of stakeholders: Developers and AI Researchers who create AI systems, Domain Experts who are specialists in the area of application, and Lay Users who come into contact with the system's functionalities.
- *Model*: Approaches to provide explanations can be classified as Model-agnostic, which apply to any model (based only on inputs and outputs), or Model-specific, which are restricted to a specific kind of model (for instance, neural networks of a specific type).

These and other aspects are discussed in greater depth in surveys such as [1, 4].

Artificial Intelligence efforts have recently evolved to hybrid systems that employ both data-driven and logical reasoning techniques. With increasing adoption of these systems, there is a need for approaches that tackle explainability as a primary consideration, fitting end user needs to specific explanation types and the system's AI capabilities. Recent works have focused on developing approaches that focus on combining data-driven and symbolic reasoning methods for capturing explainability of outputs [16, 70]. Following this idea, [19] presents an Explanation Ontology that treats explanations as a primary component of AI system design, providing a semantic representation that can help system designers to capture and structure the various components necessary to enable the generation and composition of hybrid user-centric explanations in their systems. The ontology captures several classes and properties from [66], where the authors proposed an Explanation Ontology Design Pattern to represent explanations. Note that most of the primitive styles that we adopt in this paper are captured in that ontology; incorporating other styles from that model is a possibility that is being explored for future versions of HEIST. Another interesting work is that of [47], which shows an extension of the Explanation Ontology to model explanations in the food domain. Other works in this vein include [24], in which the authors provide a hybrid explanation method based on decision trees to explain predictions that applies hidden-layer-clustering [27] on a deep neural network. Recently, [38] describes a hybrid neuro-symbolic reasoner, and their implementation in a distributed task-based framework, and explores building of deductive proofs/explanation graphs for questions regarding story understanding. Examples of other research efforts on integrating symbolic methods with Neural Networks include [40, 56].

The closest work in spirit to our approach is the work mentioned above by Chari et al. [19]; we therefore now discuss the relationship between their work and our own in more detail. Chari et al. also aim at combining hybrid explanation types by using different forms of knowledge; however, whereas their approach develops an ontology at a higher level of abstraction (which covers the AI system, the users, and the interface between the two, where the explanations lie), we on the other hand leverage ontologies at the *object level of the system*—i.e., we assume that the software being developed has a knowledge base comprised of an ontology (ABox + TBox in Description Logic or other ontological language), and explanations are built using elements from those sets as well as possibly other sources in a sub-symbolic services module. Our contributions center on the systems side, since we develop an application framework (HEIST) for knowledge-based intelligent systems that incorporate XAI capabilities; as discussed above, part of the design of HEIST incorporates primitive explanation types that are also used by Chari et al. In summary, the main difference with our proposal is that our work does not focus on introducing a user-centered general-purpose explanation approach, but rather on: (i) analyzing several explanation types for NETDER, which is an architecture designed for the development of systems capable of detecting malicious behavior in social platforms; and (ii) implementing HEIST (proposed up to the design stage in [50]), a general application framework that helps system designers build explainable socio-technical systems in any domain.

Finally, there are several similarities and differences between our work and the other approaches mentioned above. In this setting, our approach can be valuable for building systems with hybrid reasoning mechanisms capable of generating different explanation types, including combined types like the ones explored, and allowing human-in-the-loop interactions. We provide the basic building blocks for the implementation of explainable intelligent socio-technical systems based on NETDER, a particular case of the HEIST application framework, which are documented here via class and interaction diagrams and have the corresponding publicly-available source code. Despite their direct application to real-world social platforms, mechanisms to deal with network knowledge-based explanations have not been extensively studied in the literature.

6.3. XAI in Cybersecurity

Explanations are essential to decision-making support, and more importantly when these systems handle critical information (such as in cybersecurity, health, or financial systems, among others). However, XAI in cybersecurity has not received much attention in the literature; we now briefly review differences and similarities between the most salient works related to our approach.

Several topics and future challenges to be explored were discussed by Viganò and Magazzeni [69]; their focus is on stressing questions that should be taken into consideration, such as who gives/receives the explanations, what constitutes a good explanation, when the explanation should be given, how to give explanations, where the explanations should be made available, and why the explanations are needed (as discussed in Section 2). In [33], Holder et al. present a use case for capturing these issues, seeking a guide for the development of future explainable systems that users can leverage. Other works are part of a recent surge of research on enhancing the explainability of machine learning techniques; in our domain of interest in particular, Mahdavifar and Ghorbani [43] propose to do this via rule extraction from a trained DNN, which is then used to add explanation capabilities for the detection of cyber threats. In [65], Szczepanski et al. develop and evaluate an Explainable Intrusion Detection System based on a combination of ML tools via a microaggregation heuristic. A particularly closely related work is the one reported by Kuppa et al. in [39], which centers on understanding the security robustness of explainable methods by introducing a taxonomy for XAI methods, capturing various security properties and threat models relevant to the cybersecurity domain. Motivated by this, the authors then present a black-box attack to study the consistency, correctness, and confidence security properties of gradient-based XAI methods. Reviews of recent developments in the area of XAI in cybersecurity are presented in [32, 63].

Our line of research takes initial steps towards the implementation of explainable software tools based on the HEIST application framework in general, and the NETDER model in particular, providing decision support via the leveraging of both ontological knowledge and network diffusion processes. To the best of our knowledge, there have not been other works of this kind to date.

Finally, though in this paper we focus on applications related to cybersecurity, the HEIST framework (and NETDER-based implementations) can find application in many other domains that are quite far from the ones explored here, such as digital history and cultural heritage; we refer the reader to [55] and [68] for recent works in these areas. In [68], the authors state that the explainability of heterogeneous AI-driven art-related systems is a challenging and promising task notably because this functionality would allow for a better understanding of the system's outputs by artists who could also interact with the knowledge base in order to tune the performance of the underlying machine learning mechanisms. Furthermore, due to ever-increasing adoption of intelligent systems in critical fields such healthcare and finance, researchers have recognized that symbolic reasoning approaches have potential in the general effort to mitigate opacity and to make data-driven AI models more explainable [18]. There is clearly much work to be done in this direction, but preliminary results are promising.

7. Discussion: Strengths, Limitations, and Remaining Challenges

We now discuss the strengths and limitations of our work, and describe a set of challenges that arise in applying it in real-world scenarios. We begin by briefly recalling the strengths of the approach as discussed in previous sections. The main motivation behind developing a general-purpose tool like HEIST is to provide a guide for developing systems that share certain properties; in this case, we focus on intelligent socio-technical systems (ISTSs). Though HEIST assumes the existence of both a symbolic module and a set of sub-symbolic services, this does not necessarily mean that the symbolic aspect is the main one. The framework allows for the development of a range of possible types of systems; at one end of the spectrum, we may have simple rules guiding the invocation of a suite of state-of-the-art deep learning tools, while at the other we could have a Prolog-based expert system that does not use sub-symbolic services at all. The main class of problems that we aim to solve with HEIST is therefore mainly centered on the software engineering tasks that are carried out during system development. The application framework provides a structure that guides these tasks, acting as a roadmap for analysts, designers, and developers.

In terms of explainability, HEIST allows for leveraging as sources of explanations the different parts of the knowledge base (extensional, intensional, or both), the set of sub-symbolic services, as well as combinations of the two. We have included a set of primitive explanation styles in this work, and also studied how they may be combined in providing richer explanations.

7.1. Limitations

There are several limitations that can be identified in our work. The version of HEIST being presented here aims to be as general as possible, but the code that we developed in this first version has been developed as a generalization of our previous work on NetDER, which uses Datalog+/– as the underlying language; therefore, further work needs to be done in order to gather experience with other languages.

Focusing on the XAI functionalities, the current version of HEIST only supports multi-style explanations applying Policy 1 (i,e,, obtaining each style independently—cf. Section 4.3). As we have shown in our preliminary analysis of combining styles in pairs, the problem is quite challenging; developing more complex multi-style explanation delivery mechanisms requires further study. Another aspect that we have not addressed in this work is that of how to best present explanations to users depending on their *type*. In recent work [50], we have taken first steps in analyzing this problem in terms of three general types of users: basic, power, and analyst, showing possible ways in which functionalities depend on user capabilities. Such functionalities have not yet been incorporated into HEIST; this involves further developing several of the components of the model illustrated in Figure 6.

Other limitations are more closely related with the scope of the tool, given that there are many challenges associated with developing large-scale intelligent socio-technical systems. In the following we briefly describe and discuss such challenges.

7.2. Challenges

Seen from a software engineering perspective, intelligent socio-technical systems are information systems that can range in complexity from very simple (like a mobile app that applies filters to pictures and posts them to social media) to highly complex (like a monitoring system that combines feeds from many social media platforms and detects a range of possible malicious activities). There are many well-known software engineering challenges involved in building complex systems, and the level of hurdle that such challenges pose depends on many factors, such as number and types of users, number and type of data sources, impact of failures, and performance requirements, among others. In order to discuss such challenges in greater detail, we continue adopting as a general setting the cybersecurity domain, as we have done in this work and others in this line of research [48, 50, 61]—this domain exhibits all such characteristics since it covers a wide range of applications. Some of the main challenges that arise as systems become more complex along the dimensions described above are the following; note that they are often interrelated in real-world applications:

- *Data and knowledge management*: There are many challenges associated with adequately manipulating data at large scale, such as cleaning, curation, and integration. Dealing with multiple sources of potentially incomplete, inconsistent, or otherwise noisy data requires efforts that are closely related to the specific application being developed and the impact on users that decisions have. Regarding knowledge management, such activities often require the involvement of experts that are familiar with the application domains and who must work with other experts and automated tools in order to ensure that knowledge is manipulated in a principled manner.
- *Streaming data*: Though this can be considered to belong to the previous set of challenges, it deserves a separate discussion. When applied to data and reasoning, the term *streaming* refers to settings in which data is produced, transmitted, and captured at such high rates and volumes that it cannot be held for later processing; in classical databases research this is known as *stream processing*, and when additional reasoning is involved it is known as *stream reasoning*. There are many hurdles to overcome in such settings that arise from the same challenges described for data and knowledge management but at higher rates and volumes.
- *Computational decidability and tractability*: One of the issues that arise when dealing with expressive languages like existential rules and description logics is that expressive power is intrinsically tied to the compu-

tational roadblocks of undecidability and intractability. There are many research and development lines dedicated to studying such balance, and building complex systems that incorporate services based on expressive languages both adds another layer of complexity (because they become part of a greater system) and offers opportunities to explore how (un)decidability and (in)tractability conditions developed in the lab line up with conditions that arise "in the wild".

- *Ethical considerations*: The main aspect that distinguishes socio-technical systems from conventional ones is that users have an important stake in their deployment; in the case of *intelligent* systems, there are many ethical concerns that arise in relation to how sensitive data is manipulated, stored, and transmitted, how automated decisions are made and audited, how biases in training data can be avoided, and how the system and users communicate with each other.
- *Security*: The attack surface of intelligent systems is greatly expanded in comparison to traditional information systems due to many factors. Those related to classical security issues include their dependency on large sets of training data whose provenance is not always known (leading to vulnerabilities like data poisoning), functionalities that involve transmitting sensitive data (such as photos of users), granting permissions beyond those typically required to operate (such as location sharing, access to contacts, etc.), and eavesdropping on sensitive interactions. However, modern ISTSs are vulnerable to a wide range of attacks that exploit the *human component*; consider for instance how users may be affected by malicious actors on social media via cyberbullying, hate speech, and dissemination of fake news. Understanding how best to protect users from such a large range of vulnerabilities is a formidable challenge that will likely take center stage for at least the next decade of R&D all the disciplines related to ISTSs.

Though the development of effective, efficient, secure, and explainable intelligent socio-technical systems involves overcoming many hurdles, progress is steadily being made in many disciplines towards this goal.

8. Conclusions and Future Work

In this paper, we have continued a research line that began in [48] with the proposal of the NETDER architecture as a general model for reasoning about malicious behavior in social platforms and was recently generalized to the HEIST application framework for explainable query answering in cybersecurity domains, presenting its design in [50, 51]. Here, we have implemented this framework and focused on further developing the explainable query answering functionality by: (i) identifying sources of explanatory knowledge (three ontology-based and two complementary ones); (ii) identifying seven primitive explanation styles from the XAI literature plus 21 combined styles, and studying how they can be adapted to work with the source of knowledge; (iii) developing a "build-your-own explanation" approach to allow users to choose the source(s) and style(s) they want their explanation to be based on for a specific query. We developed a proof-of-concept use case based on hate speech detection in social platforms, which we used as a running example to illustrate our discussions, and extended it to a cyberbullying detection use case. Finally, we made publicly available a preliminary implementation of our application framework, and documented its use for the explanation request functionality.

There remains much work to be done as part of our efforts to materialize the vision proposed in [30] of moving "from ontology-driven *information* systems to ontology-driven *socio-technical* systems". The goal of this paper was to take a systems engineering perspective on incorporating explainability functionalities; concerns regarding the provision of adequate and high-quality explanations are outside the scope of this work since they need to be addressed for specific combinations of concrete systems and domains in order to avoid issues like the ones discussed in [6]. More concretely, building real-world intelligent socio-technical systems requires, in addition to adequate application of well-known software engineering principles, the implementation of methodologies in order to ensure the alignment between academic and industrial developments; a good example of such a methodology is described in [26]. In terms of explainability of such systems, evaluating the quality of the explanations (as proposed in [34]) that are produced is a formidable challenge since it involves many aspects that greatly transcend the already complex goal of delivering optimal user experiences. One particularly interesting aspect of measuring the usefulness of explanations to the users that consume them is related to their impact on behavior, as studied in [52]. Incorporating these issues into our research line is part of ongoing and future work.

Future work also involves studying the combination of explanation styles in greater depth in order to improve the coverage of the vast space of potential user requirements regarding explainability, continue development of our application framework to incorporate combined explanation styles, and testing its deployment in real-world applications. Finally, we are interested in exploring ways in which our work and the Explanation Ontology by Chari et al. [19] can interact in future research directions; it seems clear that, given that the two approaches are developed at different levels of abstraction, there are opportunities for each to inform the application of the other in practice.

Acknowledgments. This work was funded in part by Universidad Nacional del Sur (UNS) under grants PGI 24/ZN34 and PGI 24/N056, Secretaría de Investigación Científica y Tecnológica FCEN–UBA (RESCS-2020-345-E-UBA-REC), Universidad Nacional de Entre Ríos under grant PDTS-UNER 7066, CONICET under grants PIP 11220200101408CO and PIP 11220210100577CO, and Agencia Nacional de Promoción Científica y Tecnológica, Argentina under grants PICT-2018-0475 (PRH-2014-0007) and PICT-2020 SERIE A-01481.

Finally, we are grateful to the reviewers of this paper, who provided extensive and constructive comments that have significantly improved our work.

References

[1] A. Adadi and M. Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access*, 6:52138–52160, 2018. https://doi.org/10.1109/ACCESS.2018.2870052

[2] E. Albini, P. Baroni, A. Rago, and F. Toni. Interpreting and explaining PageRank through argumentation semantics. *Intelligenza Artificiale*, 15(1):17–34, 2021. https://doi.org/10.3233/IA-210095

[3] C. Alrabbaa, F. Baader, S. Borgwardt, P. Koopmann, and A. Kovtunova. Finding small proofs for description logic entailments: Theory and practice. In E. Albert and L. Kovács, editors, in: *Proceedings of the 23rd International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR 2020)*, volume 73 of *EPiC Series in Computing*, pages 32–67. EasyChair, 2020. https://doi.org/10.29007/nhpp

[4] A. B. Arrieta, N. D. Rodríguez, J. D. Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, 2020. https://doi.org/10.1016/j.inffus.2019.12.012

[5] A. Artelt and B. Hammer. On the computation of counterfactual explanations – A survey. *CoRR*, abs/1911.07749, 2019. https://doi.org/10.48550/arXiv.1911.07749

[6] B. Babic, S. Gerke, T. Evgeniou, and I. G. Cohen. Beware explanations from AI in health care. *Science*, 373(6552):284–286, 2021. https://doi.org/10.1126/science.abg1834

[7] V. Beaudouin, I. Bloch, D. Bounie, S. Clémençon, F. d'Alché-Buc, J. Eagan, W. Maxwell, P. Mozharovskyi, and J. Parekh. Flexible and context-specific AI explainability: A multidisciplinary approach. *CoRR*, abs/2003.07703, 2020. https://doi.org/10.48550/arXiv.2003.07703

[8] L. E. Bertossi. An ASP-based approach to counterfactual explanations for classification. In V. Gutiérrez-Basulto, T. Kliegr, A. Soylu, M. Giese, and D. Roman, editors, in:*Proceedings of the 4th International Joint Conference on Rules and Reasoning, RuleML+RR 2020*, volume 12173 of *Lecture Notes in Computer Science*, pages 70–81. Springer, 2020. https://doi.org/10.1007/978-3-030-57977-7_5

[9] A. Borgida, E. Franconi, and I. Horrocks. Explaining ALC subsumption. In W. Horn, editor, in: *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI)*, pages 209–213. IOS Press, 2000.

[10] S. Borgwardt, A. Hirsch, A. Kovtunova, and F. Wiehr. In the eye of the beholder: Which proofs are best? In S. Borgwardt and T. Meyer, editors, in: *Proceedings of the 33rd International Workshop on Description Logics (DL 2020) co-located with the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR 2020)*.

[11] M. E. B. Brarda, L. H. Tamargo, and A. J. García. Using argumentation to obtain and explain results in a decision support system. *IEEE Intelligent Systems*, 36(2):36–42, 2021. https://doi.org/10.1109/MIS.2020.3042740

[12] P. Brézillon. Context in artificial intelligence: I. A survey of the literature. *Computers and Artificial Intelligence*, 18(4), 1999.

[13] P. Brézillon. Context in artificial intelligence II. Key elements of contexts. *Computers and Artificial Intelligence*, 18(5), 1999.

[14] R. M. J. Byrne. Counterfactuals in explainable artificial intelligence (XAI): Evidence from human reasoning. In S. Kraus, editor, in: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI 2019)*, pages 6276–6282. ijcai.org, 2019. https://doi.org/10.24963/ijcai.2019/876

[15] P. Cabalar, J. Fandinno, and B. Muñiz. A system for explainable answer set programming. In F. Ricca, A. Russo, S. Greco, N. Leone, A. Artikis, G. Friedrich, P. Fodor, A. Kimmig, F. A. Lisi, M. Maratea, A. Mileo, and F. Riguzzi, editors, in: *Proceedings 36th International Conference on Logic Programming (Technical Communications), ICLP Technical Communications 2020, (Technical Communications) UNICAL*, volume 325 of *EPTCS*, pages 124–136, 2020. https://doi.org/10.4204/EPTCS.325.19

[16] R. Calegari, G. Ciatto, and A. Omicini. On the integration of symbolic and sub-symbolic techniques for XAI: A survey. *Intelligenza Artificiale*, 14(1):7–32, 2020. https://doi.org/10.3233/IA-190036

[17] A. Calì, G. Gottlob, and T. Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. *Web Semantics: Science, Services and Agents on the World Wide Web*, 14:57–83, 2012. https://doi.org/10.1016/j.websem.2012.03.001

[18] S. Chari, D. M. Gruen, O. Seneviratne, and D. L. McGuinness. Foundations of explainable knowledge-enabled systems. In I. Tiddi, F. Lécué, and P. Hitzler, editors, *Knowledge Graphs for eXplainable Artificial Intelligence: Foundations, Applications and Challenges*, volume 47 of *Studies on the Semantic Web*, pages 23–48. IOS Press, 2020. https://doi.org/10.3233/SSW200010

[19] S. Chari, O. Seneviratne, D. M. Gruen, M. A. Foreman, A. K. Das, and D. L. McGuinness. Explanation ontology: A model of explanations for user-centered AI. In J. Z. Pan, V. A. M. Tamma, C. d'Amato, K. Janowicz, B. Fu, A. Polleres, O. Seneviratne, and L. Kagal, editors, in: *Proceedings of the 19th International Semantic Web Conference (ISWC 2020)*, volume 12507 of *Lecture Notes in Computer Science*, pages 228–243. Springer, 2020. https://doi.org/10.1007/978-3-030-62466-8_15

[20] R. Confalonieri, L. Coba, B. Wagner, and T. R. Besold. A historical perspective of explainable artificial intelligence. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 11(1), 2021. https://doi.org/10.1002/widm.1391

[21] R. Confalonieri, T. Weyde, T. R. Besold, and F. M. del Prado Martín. Using ontologies to enhance human understandability of global post-hoc explanations of black-box models. *Artificial Intelligence*, 296:103471, 2021. https://doi.org/10.1016/j.artint.2021.103471

[22] K. Cranmer, J. Brehmer, and G. Louppe. The frontier of simulation-based inference. In *Proceedings of the National Academy of Sciences*, 117(48):30055–30062, 2020. https://doi.org/10.1073/pnas.1912789117

[23] A. Darwiche and C. Ji. On the computation of necessary and sufficient explanations. In *Proceedings of the 36th Association for the Advancement of Artificial Intelligence Conference (AAAI 2022)*, pages 5582–5591. AAAI Press, 2022. https://doi.org/10.48550/arXiv.2203.10451

[24] T. De, P. Giri, A. Mevawala, R. Nemani, and A. Deo. Explainable AI: A hybrid approach to generate human-interpretable explanation for deep learning prediction. *Procedia Computer Science*, 168:40–48, 2020. https://doi.org/10.1016/j.procs.2020.02.255

[25] A. K. Dey. Explanations in context-aware systems. In T. Roth-Berghofer, N. Tintarev, and D. B. Leake, editors, *Explanation-aware Computing, Papers from the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI 2009) Workshop, Pasadena, California , USA, July 11-12, 2009*, pages 84–93, 2009.

[26] M. Dragoni, I. Donadello, and C. Eccher. Explainable AI meets persuasiveness: Translating reasoning results into behavioral change advice. *Artificial Intelligence in Medicine*, 105:101840, 2020. https://doi.org/10.1016/j.artmed.2020.101840

[27] R. Féraud and F. Clérot. A methodology to explain neural network classification. *Neural Networks*, 15(1):237–246, 2002. https://doi.org/10.1016/S0893-6080(01)00127-7

[28] L. Gao and R. Huang. Detecting online hate speech using context aware models. In R. Mitkov and G. Angelova, editors, in: *Proceedings of the International Conference Recent Advances in Natural Language Processing (RANLP 2017)*, pages 260–266. INCOMA Ltd., 2017. https://doi.org/10.26615/978-954-452-049-6_036

[29] T. Grüne-Yanoff. The explanatory potential of artificial societies. *Synthese.*, 169(3):539–555, 2009. https://doi.org/10.1007/s11229-008-9429-0

[30] N. Guarino, E. Bottazzi, R. Ferrario, and G. Sartor. Open ontology-driven sociotechnical systems: Transparency as a key for business resiliency. In M. De Marco, D. Te'eni, V. Albano, and S. Za, editors, *Information Systems: Crossroads for Organization, Management, Accounting and Engineering*, pages 535–542, Heidelberg, 2012. Physica-Verlag HD. https://doi.org/10.1007/978-3-7908-2789-7_58

[31] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM Computing Survey*, 51(5):93:1–93:42, 2019. https://doi.org/10.1145/3236009

[32] S. Hariharan, A. Velicheti, A. Anagha, C. Thomas, and N. Balakrishnan. Explainable artificial intelligence in cybersecurity: A brief review. In *Proceedings of the 4th International Conference on Security and Privacy (ISEA-ISAP 2021)*, pages 1–12. IEEE, 2021. https://doi.org/10.1109/ISEA-ISAP54304.2021.9689765

[33] E. Holder and N. Wang. Explainable artificial intelligence (XAI) interactively working with humans as a junior cyber analyst. *Human-Intelligent Systems Integration*, pages 1–15, 2021. https://doi.org/10.1007/s42454-020-00021-z

[34] A. Holzinger, A. Carrington, and H. Müller. Measuring the quality of explanations: The system causability scale (SCS). *KI-Künstliche Intelligenz*, 34(2):193–198, 2020. https://doi.org/10.1007/s13218-020-00636-z

[35] M. Horridge. *Justification based explanation in ontologies*. PhD thesis, University of Manchester, UK, 2011.

[36] M. Horridge, B. Parsia, and U. Sattler. The OWL explanation workbench: A toolkit for working with justifications for entailments in owl ontologies. 2009.

[37] A. Kalyanpur. *Debugging and Repair of OWL Ontologies*. PhD thesis, University of Maryland, College Park, MD, USA, 2006.

[38] A. Kalyanpur, T. Breloff, and D. A. Ferrucci. Braid: Weaving symbolic and neural knowledge into coherent logical explanations. In *Proceedings of the Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence (IAAI 2022)*, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, pages 10867–10874. AAAI Press, 2022. https://doi.org/10.1609/aaai.v36i10.21333

[39] A. Kuppa and N. Le-Khac. Black box attacks on explainable artificial intelligence (XAI) methods in cyber security. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2020)*, pages 1–8. IEEE, 2020. https://doi.org/10.1109/IJCNN48605.2020.9206780

[40] F. Lécué, J. Chen, J. Z. Pan, and H. Chen. Knowledge-based explanations for transfer learning. In I. Tiddi, F. Lécué, and P. Hitzler, editors, *Knowledge Graphs for eXplainable Artificial Intelligence: Foundations, Applications and Challenges*, volume 47 of *Studies on the Semantic Web*, pages 180–195. IOS Press, 2020. https://doi.org/10.3233/SSW200018

[41] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis. Explainable AI: A review of machine learning interpretability methods. *Entropy*, 23(1):18, 2021. https://doi.org/10.3390/e23010018

[42] A. Lucic, H. Oosterhuis, H. Haned, and M. de Rijke. FOCUS: Flexible optimizable counterfactual explanations for tree ensembles. In *Proceedings of the Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence (IAAI 2022)*, pages 5313–5322. AAAI Press, 2022. https://doi.org/10.1609/aaai.v36i5.20468

[43] S. Mahdavifar and A. A. Ghorbani. DeNNeS: Deep embedded neural network expert system for detecting cyber attacks. *Neural Computing and Applications*, 32(18):14753–14780, 2020. https://doi.org/10.1007/s00521-020-04830-w

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

[45] E. Mumford. The story of socio-technical design: Reflections on its successes, failures and potential. *Information systems journal*, 16(4):317–342, 2006. https://doi.org/10.1111/j.1365-2575.2006.00221.x

[46] I. Nunes and D. Jannach. A systematic review and taxonomy of explanations in decision support and recommender systems. *CoRR*, abs/2006.08672, 2020. https://doi.org/10.48550/arXiv.2006.08672

[47] I. Padhiar, O. Seneviratne, S. Chari, D. Gruen, and D. L. McGuinness. Semantic modeling for food recommendation explanations. In *Proceedings of the 37th IEEE International Conference on Data Engineering Workshops, ICDE Workshops 2021*, pages 13–19. IEEE, 2021. https://doi.org/10.1109/ICDEW53142.2021.00010

[48] J. N. Paredes, G. I. Simari, M. V. Martinez, and M. A. Falappa. NetDER: An architecture for reasoning about malicious behavior. *Information Systems Frontiers*, 23(1):185–201, 2021. https://doi.org/10.1007/s10796-020-10003-w

[49] J. N. Paredes, G. I. Simari, M. V. Martinez, and M. A. Falappa. Detecting malicious behavior in social platforms via hybrid knowledge- and data-driven systems. *Future Generation Computer Systems*, 125:232–246, 2021. https://doi.org/10.1016/j.future.2021.06.033

[50] J. N. Paredes, J. C. L. Teze, M. V. Martinez, and G. I. Simari. The HEIC application framework for implementing XAI-based socio-technical systems. *Online Social Networks and Media*, 32:100239, 2022. https://doi.org/10.1016/j.osnem.2022.100239

[51] J. N. Paredes, J. C. L. Teze, G. I. Simari, and M. V. Martinez. On the importance of domain-specific explanations in AI-based cybersecurity systems (Technical Report). 2021. https://doi.org/10.48550/arXiv.2108.02006

[52] M. Poveda-Villalón, A. Fernández-Izquierdo, M. Fernández-López, and R. García-Castro. LOT: An industrial oriented ontology engineering framework. *Engineering Applications of Artificial Intelligence*, 111:104755, 2022. https://doi.org/10.1016/j.engappai.2022.104755

[53] M. T. Ribeiro, S. Singh, and C. Guestrin. Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM, 2016. https://doi.org/10.1145/2939672.2939778

[54] M. J. Robeer. Contrastive explanation for machine learning. Master's thesis, Utrecht University, 2018.

[55] N. D. Rodríguez and G. Pisoni. Accessible cultural heritage through explainable artificial intelligence. In T. Kuflik, I. Torre, R. Burke, and C. Gena, editors, in: *Adjunct Publication of the 28th ACM Conference on User Modeling, Adaptation and Personalization (UMAP 2020)*, pages 317–324. ACM, 2020. https://doi.org/10.1145/3386392.3399276

[56] M. K. Sarker, N. Xie, D. Doran, M. L. Raymer, and P. Hitzler. Explaining trained neural networks with Semantic Web technologies: First steps. In T. R. Besold, A. S. d'Avila Garcez, and I. Noble, editors, in: *Proceedings of the Twelfth International Workshop on Neural-Symbolic Learning and Reasoning (NeSy 2017)*, volume 2003 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2017.

[57] R. K. Sawyer. Social explanation and computational simulation. *Philosophical explorations*, 7(3):219–231, 2004. https://doi.org/10.1080/1386979042000258321

[58] M. R. G. Schiller and B. Glimm. Towards explicative inference for OWL. In T. Eiter, B. Glimm, Y. Kazakov, and M. Krötzsch, editors, in: *Informal Proceedings of the 26th International Workshop on Description Logics*, volume 1014 of *CEUR Workshop Proceedings*, pages 930–941. CEUR-WS.org, 2013.

[59] A. Schofield and T. Davidson. Identifying hate speech in social media. *XRDS: Crossroads, The ACM Magazine for Students*, 24(2):56–59, 2017. https://doi.org/10.1145/3155212

[60] P. Shakarian, G. I. Simari, and D. Callahan. Reasoning about complex networks: A logic programming approach. *Theory and Practice of Logic Programming*, 13, 2013. https://doi.org/10.48550/arXiv.2209.15067

[61] G. I. Simari. From data to knowledge engineering for cybersecurity. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI 2019)*, pages 6403–6407, 2019. https://doi.org/10.24963/ijcai.2019/896

[62] I. Sommerville. Software engineering, 10th edition (international computer science). *Essex, UK: Pearson Education*, pages 1–808, 2016.

[63] G. Srivastava, R. H. Jhaveri, S. Bhattacharya, S. Pandya, Rajeswari, P. K. R. Maddikunta, G. Yenduri, J. G. Hall, M. Alazab, and T. R. Gadekallu. XAI for cybersecurity: State of the art, challenges, open issues and future directions. *CoRR*, abs/2206.03585, 2022. https://doi.org/10.48550/arXiv.2206.03585

[64] I. Stepin, J. M. Alonso, A. Catalá, and M. Pereira-Fariña. A survey of contrastive and counterfactual explanation generation methods for explainable artificial intelligence. *IEEE Access*, 9:11974–12001, 2021. https://doi.org/10.1109/ACCESS.2021.3051315

[65] M. Szczepanski, M. Choras, M. Pawlicki, and R. Kozik. Achieving explainability of intrusion detection system by hybrid oracle-explainer approach. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2020)* , pages 1–8. IEEE, 2020. https://doi.org/10.1109/IJCNN48605.2020.9207199

[66] I. Tiddi, M. d'Aquin, and E. Motta. An ontology design pattern to define explanations. In K. Barker and J. M. Gómez-Pérez, editors, in: *Proceedings of the 8th International Conference on Knowledge Capture (K-CAP 2015)*, pages 3:1–3:8. ACM, 2015. https://doi.org/10.1145/2815833.2815844

[67] J. Uyheng and K. M. Carley. Characterizing network dynamics of online hate communities around the COVID-19 pandemic. *Applied Network Science*, 6(1):20, 2021. https://doi.org/10.1007/s41109-021-00362-x

[68] E. van der Peijl, A. Najjar, Y. Mualla, T. J. Bourscheid, Y. Spinola-Elias, D. Karpati, and S. Nouzri. Toward XAI & human synergies to explain the history of art: The smart photobooth project. In D. Calvaresi, A. Najjar, M. Winikoff, and K. Främling, editors, in: *Proceedings of the Explainable and Transparent AI and Multi-Agent Systems: Third International Workshop (EXTRAAMAS 2021)*, volume 12688 of *Lecture Notes in Computer Science*, pages 208–222. Springer, 2021. https://doi.org/10.1007/978-3-030-82017-6_13

[69] L. Viganò and D. Magazzeni. Explainable security. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 293–300. IEEE, 2020. https://doi.org/10.1109/EuroSPW51379.2020.00045

[70] L. Yang, J. Liu, F. Ye, Y. Wang, C. D. Nugent, H. Wang, and L. Martínez. Highly explainable cumulative belief rule-based system with effective rule-base modeling and inference scheme. *Knowledge-Based Systems*, 240:107805, 2022. https://doi.org/10.1016/j.knosys.2021.107805

[71] Q. Zhong, X. Fan, X. Luo, and F. Toni. An explainable multi-attribute decision model based on argumentation. *Expert Systems with Applications*, 117:42–61, 2019. https://doi.org/10.1016/j.eswa.2018.09.038

Appendix A. Additional material

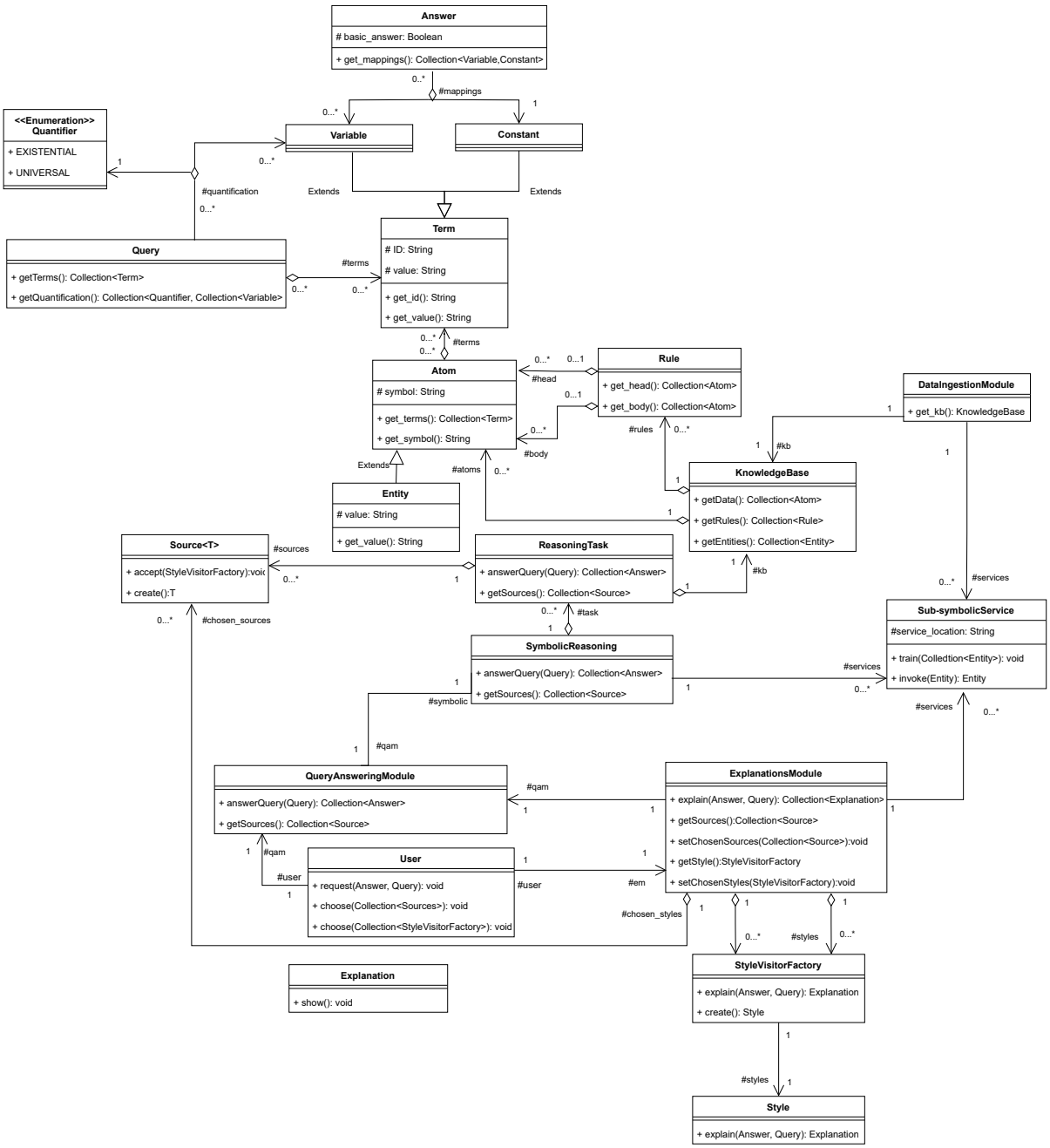

Fig. 9. Detailed class diagram (low-level view of Figure 6).