Enhancing Explainability in Knowledge Graph Construction for Healthcare Services Using Large Language Models

Andrea Molinari[1, 2] and Simone Sandri[3]

[1] Dept. Of Industrial Engineering, University of Trento, Italy
[2] Lappeenranta University of Technology, Finland

Andrea.molinari@unitn.it
[3] Okkam s.r.l., Trento, Italy
sandri@okkam.it

Abstract. The widespread deployment of intelligent systems in critical sectors—such as healthcare, justice, and finance—has underscored the urgent need for transparent and trustworthy in the answers provided. Accordingly, explainability constitutes a key requirement to foster understanding, trust, and accountability in automated decision-making processes where AI in general, and Large Language Models (LLMs) in particular, are involved. This paper introduces a new explainability framework for symbolic systems, centered on an application pipeline aimed at constructing a semantic knowledge graph from unstructured text. In our approach, explainability for the symbolic entity recognition component is achieved through an event-driven system, where each operation is logged atomically in a journal, capturing events that trace object creation, lineage, and purpose. Alongside the journal, two additional components support the architecture: the Entity Storage, which maintains versioned records of entities, and the Explainability Module, which interprets and explains the logic behind each entity's representation. The proposed framework offers a high degree of detail and customizability and is currently being tested within a medical application designed for general practitioners. In this context, the ability to provide transparent justifications for specific recommendations is essential to ensuring reliability for both physicians and patients. Furthermore, it contributes to strengthening the credibility and trustworthiness of AI-based tools, which are expected to assume an increasingly central role in this domain.

Keywords: Explainability, Symbolic Systems, Knowledge Graph Construction, Patient Information Leaflet.

1 Introduction

In today's rapid widespread adoption of artificial intelligence (AI) technologies and advanced language models such as ChatGPT, the issue of explainability has transcended its original framing as a regulatory requirement—most notably, for Europe,

under the General Data Protection Regulation (GDPR). It has now emerged as a critical concern tied to ethical accountability, organizational transparency, and stakeholder trust. Across both public and private sectors, the absence of clear and accessible mechanisms for explaining AI-driven processes and decisions is increasingly viewed as a fundamental shortcoming in the responsible deployment of AI. This concern is especially serious in high-stakes contexts such as healthcare, judicial decision-making, and public administration—domains where AI-supported decisions can directly affect citizens' rights, safety, and well-being.

In this paper, we introduce a framework designed to provide fine-grained explainability in AI applications, with a particular focus on knowledge graph construction and entity matching within text-to-graph transformation processes. The framework was developed as part of the M.A.E.S.T.R.O. project, which aims to address a central limitation of large language models (LLMs): their propensity to generate hallucinations, that is, outputs that are plausible yet factually incorrect or fabricated. This issue presents a significant obstacle to the adoption of LLMs in high-stakes environments such as healthcare, legal systems, and scientific research. Our approach tries to mitigate this risk by developing strategies for extracting, verifying, and presenting information in a manner that promotes factual accuracy and reduces hallucination rates. To this end, we integrate symbolic reasoning, knowledge graph methodologies, and fine-tuned extraction pipelines to enhance the reliability and trustworthiness of LLM-generated outputs.

Explainability in our approach is operationalized through a dedicated, event-driven architecture embedded within the symbolic entity recognition component. This ensures interpretability of the inferred structures and relationships drawn from domain knowledge—in this case, medical indications pertaining to patients. The modular architecture systematically logs and contextualizes all symbolic operations performed during both network construction and entity recognition phases. At the heart of this architecture lies a Journal component, which records an immutable sequence of atomic events—each documenting operations such as node creation, edge formation, attribute enrichment, and community assignment. These events are logged locally and then ingested into an Elasticsearch ™ index via Logstash ™, enabling detailed inspection and traceability of every transformation applied to the graph. In parallel, an Entity Storage Module called NEMS (Named Entities Management System) maintains versioned snapshots of graph elements, allowing analysts to track the evolution of nodes and edges over time. Finally, the Explainability Module translates these events into human-readable justifications, supporting both technical audits and user-facing transparency. This infrastructure enables stakeholders to trace the origins of any network pattern or emergent concept / entity, understand the computational steps that led to its formation, and verify its dependence on the original input data and algorithmic choices. Such capabilities are increasingly essential in both research and policy contexts, where opaque, black-box models are no longer acceptable. In particular, the growing use of social network analytics demands tools that are not only powerful but also interpretable and accountable.

The paper is structured as follows: Section 2 provides an overview of the current state of the art in AI explainability, Section 3 examines the evolution of explainable AI,

with a focus on the shift from black-box to symbolic models and Section 4 presents our explainability framework in detail, highlighting its core implementation features and discussing key findings and open challenges.

2 Explainability: motivations and state-of-the-art

Explainability in an IT system refers to the system's ability to provide clear and understandable explanations for its decisions or predictions. In other words, a system that incorporates an explainability model not only produces outcomes but is also capable of justifying them in a way that is interpretable by humans, offering a set of reasons underlying the choices made. This capability is particularly critical in domains where decisions have a significant impact on people's lives, such as healthcare, justice, and finance. The fields of application for explainability models are numerous, and the importance of being able to attribute meaning to the results can be justified in various ways [1]. The explainability model presented in this paper takes inspiration from a currently running project call "M.A.E.S.T.R.O.", whose translation means "mitigating hallucinations in large language models: optimized information extraction ".

M.A.E.S.T.R.O. aims to enhance the deep understanding of textual content and simultaneously improve the accuracy and clarity of analytical outcomes. This is achieved through the innovative integration of a symbolic approach—based on formal categorization and extraction rules and an ontology manually validated by expert Knowledge Engineers—with the latest neural and non-neural Deep Learning and Machine Learning algorithms. In essence, the project seeks to develop a hybrid AI approach (hereafter, Hybrid AI), capable of supporting applications that leverage the strengths of two typically divergent paradigms: statistical correlation and deep semantic understanding. While often positioned as alternatives, their integration within a unified platform represents a promising evolution for AI as a whole and could serve as a game-changer for NLP, particularly for organizations unable to compete with dominant global players. The overarching goal is to enhance the reliability of AI-driven, customized services for clients by improving the system's ability to: a) detect hallucinations—the generation of non-existent, incorrect, or misleading content by generative AI models, which often remain difficult to identify promptly; b) contribute to the ongoing efforts of the European Union to establish a distinct European approach to AI (see the AI Act, June 2023), emphasizing a Human-in-the-Loop (HITL) paradigm and mitigating the alienating and oracle-like behavior of Large Language Models (LLMs).

While it is unrealistic to expect a project like MAESTRO to address all structural challenges of generative AI, it is both feasible and meaningful to tackle a specific and high-value task—one still underexplored in current research: the automatic recognition and extraction of named entities (Named Entity Recognition – NER), specifically in the context of pharmaceutical texts. To this end, MAESTRO proposes a solution that is more transparent, sustainable, practical, and human-centered than those currently offered by global tech giants. It seeks to make explicit the reasoning steps underlying automated analyses. In particular, it addresses the issue known in the field of generative AI as the "chain of thought", which refers to the logical sequence of steps that data are

assumed to follow within the neural network and the embedding vectors representing the original dataset. This chain ultimately determines the final output (the prediction).

This paper focuses on one of the principal aims of the project, i.e., to improve entity matching processes in knowledge graph creations by comparing a traditional approach based on entity-matching algorithms with the obscure but efficient entity identification mechanism used by modern LLMs like ChatGPT, Claude, Llama etc. These tools are providing phenomenal results when applied to contexts where web information are available, but evidently lack of this capability in specific domains where, for example, documents are private and not available to their scraping mechanisms. Secondly, LLMs are subject to the famous problem of “hallucinations”, even in recent times the situation is improving, but not solved completely. In the context of LLMs, "hallucinations" refer to the generation of answers by the model that are factually incorrect, fabricated, or nonsensical, even though it may appear fluent and plausible.

Hallucinations occur when an LLM produces information that is not grounded in its training data or external knowledge, leading to inaccurate or misleading outputs. There are two main types of hallucinations: a) factual hallucinations, where the model states false information as if it were true (e.g., inventing non-existent articles, events, or citations) and b) logical hallucinations, where the LLM draws invalid or illogical conclusions, even if all the input data is accurate. In both situations, hallucinations are particularly critical in domains like medicine, law, or education, where trust, accuracy, and verifiability are essential. In these domains, and specifically in the healthcare domain where our project is grounded, the LLM approach and interface is of great help, but it is not possible or recommendable to use mainstream solutions. This is the core business of solutions that we are developing in our research project: provide

As AI algorithms become increasingly integrated into various aspects of society, there is a growing demand from diverse stakeholders for these systems to provide interpretable and transparent outputs [13], including government regulators, impacted individuals, domain experts, and developers, In response to these diverse needs, explainability needs not only institutional constraints, but also software toolkits that incorporate state-of-the-art explainability techniques and modular architectures designed to align these methods with different stages of any AI pipeline like the one proposed in this paper.

The first characteristics of an explainability model are transparency and Trust. Users should understand how and why certain decisions are made by the AI solution, thereby increasing trust in the system. For instance, in the healthcare sector, it is essential for professionals to comprehend the rationale behind an automated diagnosis in order to effectively integrate it into the clinical decision-making process. At the same time, patients can also benefit from having clear and understandable diagnoses. Secondly, the model should provide identification mechanisms of bias and errors. By analyzing the process of the knowledge graph generation from the original texts, it becomes possible to detect and correct inaccuracies or misalignments in the nodes and relationships generation, or in the entity identification. For example, the use of web-based biased data or an unfair comments on a medical usage can lead to significant errors during AI model generation[12], as the models become influenced by non-representative and unsuitable input data. A well-known study [2] revealed that a widely

used predictive algorithm in the United States, aimed at identifying and supporting patients with complex healthcare needs, exhibited a significant racial bias: given identical clinical conditions, black patients were found to be substantially sicker than their white counterparts.

- Regulatory Compliance: In many sectors, it is legally required that automated decisions be explainable and justifiable. Regulations such as the GDPR in Europe mandate that organizations provide clear explanations for automated decisions that affect individuals.
- Continuous Improvement: Understanding the decision-making process enables the continuous optimization and refinement of both symbolic algorithms and AI models. For example, techniques such as LIME (Local Interpretable Model-agnostic Explanations) allow for the analysis of how small changes to the input affect predictions, thereby facilitating model enhancement.

Explainability models can generally be categorized into two broad groups: those applied to symbolic or rule-based approaches, and those aimed at explaining the outcomes of machine learning models. The former—such as rule-based systems, formal logic frameworks, or declarative models—are inherently more transparent, as their internal logic is explicit and human-readable. Nonetheless, advanced techniques can be employed to further enhance their explainability and comprehensibility, particularly in complex settings or when integrated with more opaque components, such as in hybrid models. The latter group includes systems that pose significant challenges in terms of interpretability and transparency, especially complex predictive models such as recurrent neural networks (RNNs). Many AI models, especially those based on neural architectures, are intrinsically complex and opaque, making their decision-making processes difficult to interpret and understand. Beyond intrinsic model complexity, additional obstacles hinder AI explainability, such as scalability. Moreover, eXplainable AI (XAI) raises ethical concerns related to transparency and privacy, as efforts to interpret model decisions may inadvertently expose sensitive information or result in data leakage.

Issues related to AI explainability are particularly pronounced in the medical domain, which is arguably one of the first sectors to have raised concerns about this topic. On one hand, increasingly broad areas of medicine have begun adopting recommendation systems, with large language models (LLMs) specifically trained on medical contents (like the project M.A.E.S.T.R.O.) offering a substantial increase in potential. On the other hand, it is becoming increasingly evident that, given the quality of responses these models can produce—even in the context of relatively simple medical queries—the outcomes for both lay users and healthcare professionals can be remarkably effective. Nevertheless, the risk associated with their use by non-specialized personnel remains significant. Even when such tools are employed by medical professionals—a scenario that is now widespread—the problem of hallucinations, and consequently the need for verification and explainability, remains critical. An excellent analysis of the challenges surrounding explainability in the medical field can be found here. [14]

3 Explainable AI: from black boxes to symbolic models

When designing an explainability model, it is essential to ask what is actually meant by the term "explain." In the field of Artificial Intelligence (AI), the term explanation is often used interchangeably with related concepts such as interpretability, comprehensibility, and transparency. However, despite recent efforts [3], a universal and widely accepted definition is still lacking. Gomes et al. [4] further emphasize that explainability is not a monolithic concept but rather one that varies depending on the target audience. AI engineers, for instance, require explanations that clarify the model's architecture and behavior in order to modify or optimize it. Domain experts need explanations that replicate or approximate their own reasoning processes. End users demand high-level explanations that foster trust, while ethicists and policymakers require demonstrations of fairness, safety, and reliability.

Explainability in black-box models is generally addressed through post-hoc techniques—methods applied after model training to interpret or explain its behavior. In other words, a post-hoc method takes an already developed model (e.g., a deep neural network, random forest, or support vector machine) and applies techniques to provide human-understandable explanations of why a specific prediction or decision was made. Among the most influential and widely adopted post-hoc methods are LIME, SHAP, PDP and ALE.

LIME (Local Interpretable Model-agnostic Explanations) [5] is one of the most well-known methods in the field of AI model explainability, developed by Ribeiro, Singh, and Guestrin in 2016. The key feature of LIME is its ability to provide interpretable explanations for complex models (such as neural networks or ensemble methods) by approximating them locally with simpler, interpretable models—such as linear regression or decision trees. In this way, LIME offers a local perspective on model decisions, allowing users to understand how and why a complex model made a specific prediction. SHAP (Shapley Additive Explanations) [6] is, in our opinion, one of the most advanced and mathematically grounded techniques for explaining complex AI models. SHAP is based on the concept of Shapley values from cooperative game theory. In essence, SHAP explains how each feature contributes to a given prediction, ensuring a fair and consistent attribution of feature importance. Partial Dependence Plots (PDP) [7] is a widely used tool in the domain of Explainable AI (XAI), designed to visualize the marginal effect of one or more input features on the prediction of a machine learning model. These plots show how the predicted outcome changes as a specific feature varies, while all other features are held constant. Accumulated Local Effects (ALE) [8] is similar to PDPs but better suited for handling correlated features. ALE plots illustrate how a model's output changes in response to variations in a particular feature by aggregating local effects across the input space, offering a more reliable interpretation in the presence of feature dependencies.

Explainability techniques for rule-based (or symbolic) models belong to a fundamentally different category compared to the black-box approaches previously discussed: in this case, the model itself is inherently interpretable, as the decision-making process is explicit and constructed through symbolic rules, logics, or if-then statements. However, even for these systems, effectively communicating the reasoning

to human users—especially non-experts—can still be challenging, particularly due to the complexity of the operations performed on the input data.

One fundamental technique to enhance explainability in symbolic systems is rule tracing, which involves highlighting which rules (and in what sequence) have been applied to reach a specific conclusion. This technique was successfully implemented in the MYCIN system [9], one of the earliest and most renowned examples of a symbolic expert system, developed in the early 1970s at Stanford University. MYCIN could provide interactive natural language explanations in response to queries such as: "Why?" → Why are you asking me for this information?; "How?" → How did you reach this conclusion? "What if?" → What would happen if I changed an input?

Another explainability technique for symbolic models involves logical justifications. This method provides a backward explanation of the conclusion, similar to a logical proof, where each step is motivated by an activated rule and a known premise. The justification often takes the form of a chain of "because" statements: "I made this decision because rule X was triggered by these conditions." This form of explainability has also been studied within the field of Argumentation Theory [10], where conclusions are presented as part of a rational argument (comprising support, claim, and rebuttal).

4 Using LLMs to improve entity matching in knowledge graphs: impletementation and discussion

One of the selected application domains for testing the explainability model presented in this paper is the use of AI chatbots to support physicians in the prescription of medications. This scenario was investigated through the ingestion of approximately 5,000 patient information leaflets (PILs) currently used within the Italian healthcare system, freely available in PDF format. Leveraging this domain-specific, scientifically validated knowledge base—one that is not influenced by social or external biases, at least at this stage of the project—we aimed to construct an entity-centric, ontology-driven knowledge graph. This graph serves as the foundation for a chatbot system designed to provide prescription support to medical professionals. The physician is thus assisted by a chatbot trained on the PIL corpus, where the identification of key entities (e.g., drugs, symptoms, diseases, contraindications) relies on an entity matching pipeline developed through domain-specific Named Entity Recognition (NER) techniques.

The identification of entities during the construction of the entity-centric knowledge graph will be compared with entities extracted in parallel by a Large Language Model—currently LLAMA, due to its open-source availability. Entities obtained through these two parallel pipelines (i.e., a domain-specific NER process and LLM-based entity extraction) will be matched through a semi-automated entity matching procedure. In cases of perfect alignment, or differing identification above a predefined confidence threshold, the entity will be stored within a custom-developed entity management system (NEMS – Named Entity Management System). For entities falling within a gray area—i.e., not matching within the confidence threshold—a manual disambiguation process will be triggered. Candidate entities will be submitted for

validation by a domain expert (physician), who will confirm which candidate best represents the intended concept. The full process is represented in figure 1.

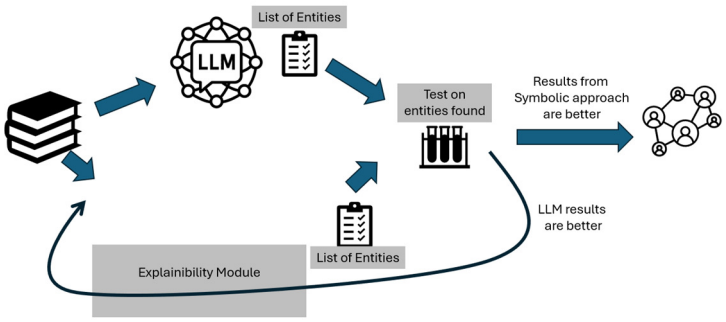

Fig 1: Entities extracted by LLM vs entities extracted with a symbolic approach

Depending on the specific configuration adopted, the explainability system presented in this paper will assist system administrators in identifying, through a backtracking process, which components of the entity matching algorithms require improvement, the underlying reasons for incorrect outputs, and which parametric values within the Graph Neural Network (GNN) used during the entity matching phase should be adjusted. For further details on the use of GNNs in Named Entity Recognition (NER) processes, see [11]. Below is an ideogram representing the parallel process of entity extraction, the comparison of results, and the intervention of a domain expert in cases of conflicting entity matching outcomes between the Large Language Model (LLM) and the proprietary NER pipeline. Specifically, the explainability module introduced in this work is integrated within the proprietary NER process and is composed of five distinct phases, as illustrated in figure 2:

1. Linguistic Features Extraction: through the process of linguistic features extraction, we identify relevant text elements and the involved relations
2. Graph Building: a dedicated algorithm based on linguistic features (LF2G) transforms tokens and lemmas into nodes of an oriented graph
3. Named Entity Recognition: GNNs are used to recognize the semantics of an entity based on adjacent nodes and the type of relation between each other
4. Entity Matching: the Entity Name System [18] will be able to identify which nodes of the graph belong to the same entity.
5. Graph Merge + ENS Enrichment: the ENS will identify which nodes of the graph belong to the same entity.

At the end of these processes, a knowledge graph is created and refined for each document (figure 3): the resulting graphs of each document are (conceptually) merged into a unique knowledge graph, ready for being browsed, queried or ready for infering and persist new knowledge. An ontology developed for the specific domain is used to refine results. Within this context lies the explainability module proposed in this work; it impacts each of the five phases with a high level of granularity, enabling us to identify where the knowledge graph construction process can be improved through comparison with the outputs generated by the LLMs.

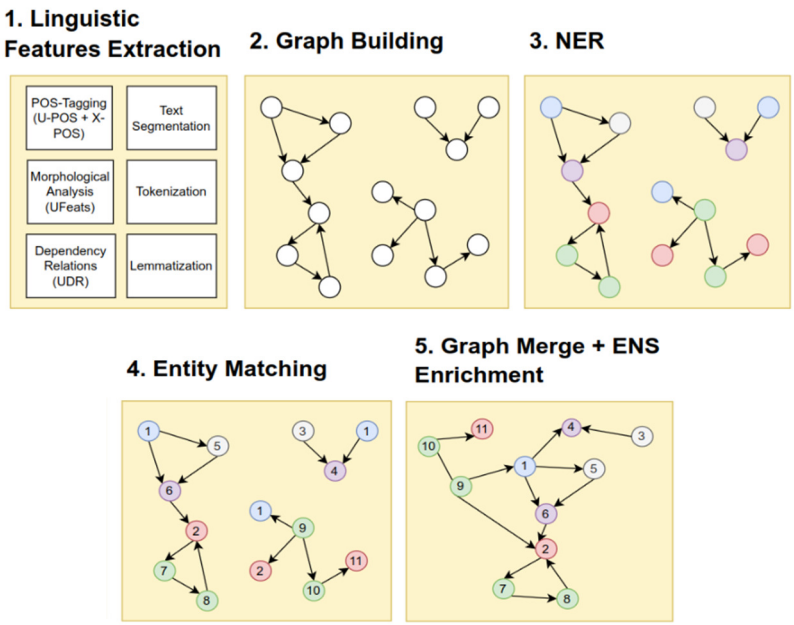

Fig 2: the process of knowledge graph creation

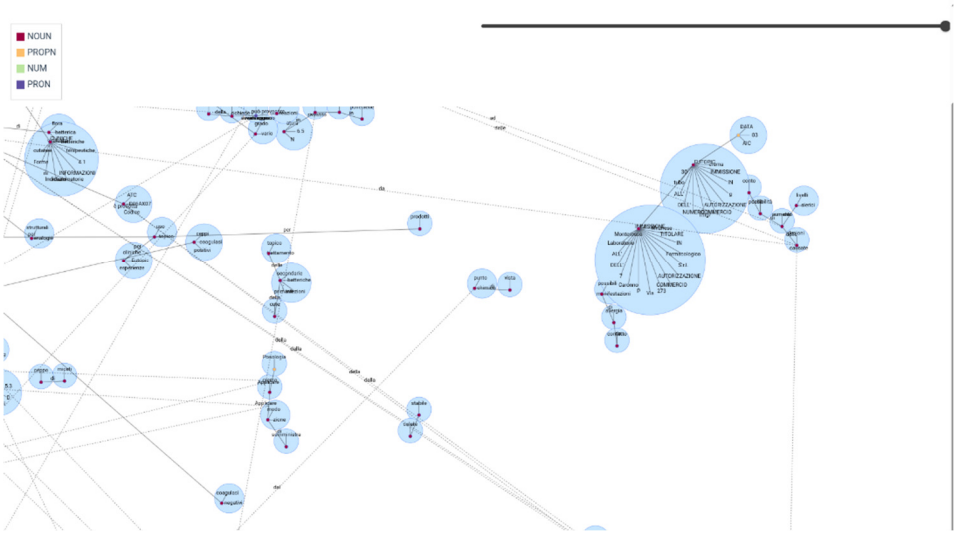

Fig 3: the knowledge graph created for the Patient Information Leaflet for Doxycycline

Explainability within the approach we have developed is ensured, for the symbolic entity recognition component, by an event-driven system that follows the techniques

outlined in the previous chapter. Specifically, every operation performed within the symbolic system passes through a Journal that atomically and selectively captures a series of events describing the actions carried out. This mechanism allows for maintaining a detailed history for each object, documenting how it was created, from which preceding object it was generated, and for what reason. In addition to the Journal, which collects the event logs, two further components complete the architecture: the Entity Storage, responsible for maintaining a versioned record of individual objects, and the Explainability Module, which is capable of describing—and thus explaining—the rationale behind the specific representation of entities.(Figure 4)

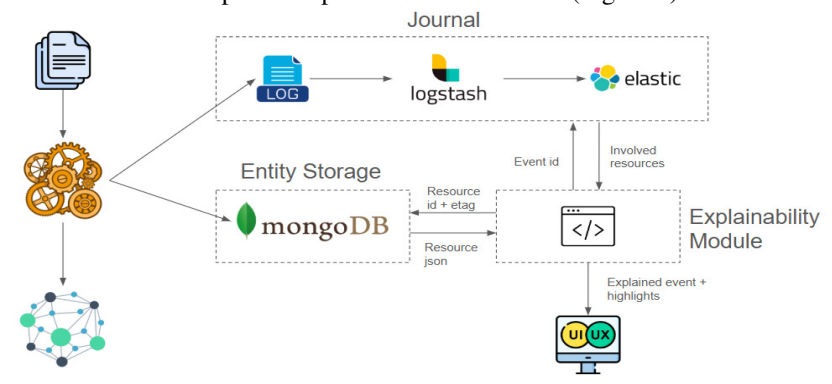

Fig.4 Ideogram of the presented explainability model

As previously described at a high level, the Journal is the component responsible for tracking all operations performed within the system. This results in a collection of logs that record a sequence of events. These logs are written to files within the local filesystem where the application runs and are periodically scanned by Logstash for indexing into Elasticsearch, which serves as the event repository. The events are incremental and evolutionary in nature; they are not subject to modification or deletion, thereby ensuring proper lifecycle management of the elements constituting each event. At a more detailed level, the Journal is implemented as an integral part of the overall process: given a specific entry representing a typed event, it leverages a logger to write the data to file. In order to ensure proper management of log files—which are only intended to be persisted temporarily before their eventual indexing into Elasticsearch, and which can grow significantly over time—a rolling mechanism (or file rotation) has been implemented. This mechanism allows the system to work with consistently small-sized files and enables the deletion of data that has already been processed and indexed. The rolling mechanism is governed by the configuration described below, which specifies directives to generate log files with a maximum size of 10 MB or containing log data no older than one day. Each individual file is deleted 30 days after its last modification date.

The log files generated in this way are permanently persisted into Elasticsearch with the support of Logstash. Logstash is responsible for reading each line of the log file and interpreting it, using a Grok expression, as a distinct object to be indexed into a specific index within the Elasticsearch cluster. Only the lines that conform to a predefined

format are considered, as specified in the excerpt of the configuration file shown below. The fields of the Elasticsearch document to be indexed are populated according to the type and extraction directives defined within the Grok expression. All atomic operations involving elements relevant to explainability are collected through a series of events, which are recorded in a log file. The format of these events is described by the following fields:

- Id → A unique identifier used for direct querying of the Journal;
- OperationType → Describes the type of operation that was executed. This information is particularly important for explainability purposes, as it defines what happened during the event;
- InputResources → A list of resources that were involved as inputs in the event;
- OutputResources → A list of resources resulting from the operations described in the event;
- InvolvedResources → The complete list of resources involved in the operation. In addition to input and output resources, it includes all auxiliary resources relevant to the operation (e.g., semantic rules, matching decisions, etc.).
- TriggerType → Describes the reason that triggered the operation. From an explainability perspective, this field defines why a resource was created or modified during the event;
- Timestamp → date and time of the performed operation, enabling chronological ordering of all events related to a specific resource, and therefore the reconstruction of the full chain of operations from the resource's creation to its current state;

It is important to emphasize that, within the Journal, resources are managed as versioned entities. Therefore, it is possible for the same resource to appear both as an input and as an output, reflecting a version update. Resource versions are always incremental. The types of operations to which the events refer are described in the following table: they reflect all possible actions involved in the pipeline that transform text into a graph and during the subsequent management of the created entities:

NODE_CREATED	A new node has been created within the graph;
NODE_ADDED	A node originating from another graph has been added to the current graph;
NODE_STRUCTURED	A new node has been structured within the graph;
NODE_MERGED	Two or more nodes have been merged into a single node. Information previously distributed across the source nodes is now consolidated into the single target node;
NODE_SPLITTED	A node has been split into two or more distinct nodes. The information from the original node has been redistributed across the new destination nodes;
NODE_ENRICHED	A node has been enriched with new data properties;
NODE_CONNECTED	Two nodes have been connected via a previously created relationship (edge);
NODE_EXTRACTED	A new node has been extracted and created within the graph;
EDGE_CREATED	A new relationship (edge) has been created and is ready to be connected;
EDGE_ENRICHED	A relationship has been enriched with new contextual information;

MATCH	A match between two entities has been detected;
ENTITY_MERGED	Two or more entities have been merged into a single entity.

Table 1: types of operations to which the events refer

Alongside the operation type referenced in each event, there is also information regarding the triggers that caused the event. As with the operation types, the following list represents the current state and may be subject to future changes (table 2):

RULE_TRIGGERED	In the symbolic context, the event's target operation was triggered by the activation of a specific rule applied during the navigation of the text's dependency tree;
OKKAM_TREE_PARSED	The target operation originated from the parsing of the text's constituency tree, detecting token pairings through the resolution of anaphora and cataphora;
GRAPH_MERGED	The target operation occurred because of merging two separate graphs into a single knowledge graph.

Table 2: triggers classification

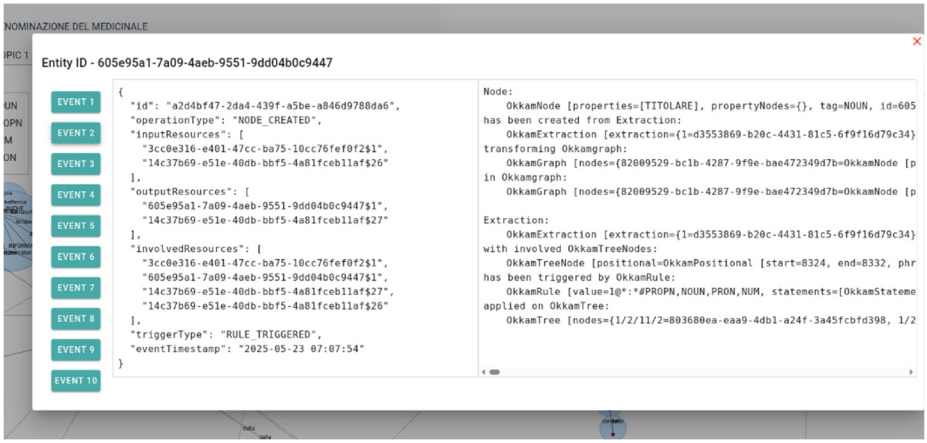

Fig. 5: Example of event collection related to the

While events associated with individual resources passing through the system are persisted into Elasticsearch, the resources themselves are stored in the Resource Storage to enable future queries. Due to the dynamic and inherently unstructured nature of the resources, the use of a relational database is not suitable. Instead, a NoSQL database is required to provide greater flexibility. For this reason, we chose MongoDB as the persistence layer for resources, allowing us to store BSON documents with heterogeneous formats and contents within the same collection—referred to as history for consistency—without the need to explicitly define or enforce a rigid data schema. To ensure proper and simplified querying of the MongoDB collection, each resource that is persisted extends a common class containing three fields that are essential for resource search and unique identification:

- ID → Represents the unique identifier of the resource;

- ETAG → A sequential number identifying the resource version. Multiple documents within the history collection may share the same "Id" field—thus referring to the same resource—but their "Etag" must, by design, differ, thereby identifying different versions. The incremental nature of the "Etag" field enables accurate reconstruction of the resource's evolution;
- RESOURCETYPE → the type of resources managed are the following

```
public enum ResourceType {

    TOKEN, NODE, EDGE, EXTRACTION, OKKAM_TREE, OKKAM_TREE_NODE, OKKAM_EXTRACTION_RULE, OKKAM_GRAPH,
    OKKAM_CONSTITUENCY_TREE, OKKAM_CONSTITUENCY_TREE_NODE, DOCUMENT, OKKAM_MENTION

}
```

Fig. 6: list of resource types currently used in the explainability module

The Explainability Module is responsible for interacting with events and resources to reconstruct the historical evolution and make results understandable and interpretable. Given a specific resource, the module can collect all events related to that resource by searching for all events where the resource appears in the outputResources field, and then chronologically ordering them to reconstruct the resource's temporal evolution. Once all the resources that contributed to an event have been collected, the Explainability Module translates this information into a human-readable format, based on the type of operation that was performed and the trigger that caused it.

At the current stage of the M.A.E.S.T.R.O. project, we are evaluating explainability by analyzing logs derived from small- to medium-sized graphs, built from texts of up to four pages—the average length of patient information leaflets (PILs) in Italy. To limit log file growth during this experimental phase and facilitate verification using HITL, we have segmented the texts into chunks ranging from 20 to 25 tokens (average italian token ≈ 5.4 characters, but for medical domain could be considered around 7 characters), selecting entire, semantically coherent paragraphs. This approach has resulted in log files averaging around 10kB of data, corresponding to an approximate ratio of 1:60. The file format used for the explainability module—described in detail later in this paper—is currently highly verbose and can undoubtedly be optimized in terms of both structure and storage efficiency. Nonetheless, it already provides valuable insights into how granular the logging operations should be, and highlights the need to assess the trade-off between log granularity and the overall quality of the explainability produced

5 Conclusions

In this work, we have examined the critical importance of explainability in both symbolic systems and AI-based systems. Explainability not only enhances user transparency and trust in automated systems but also enables the identification and correction of data biases, the continuous improvement of models, and the fulfillment of regulatory requirements. Ensuring that AI-driven recommendations and actions are

interpretable and explainable is no longer optional, but a fundamental prerequisite for legitimacy, acceptance, and social sustainability. Organizations that fail to address this aspect risk eroding stakeholder confidence, facing reputational damage, and encountering barriers in the implementation of AI-enabled services, especially in public sector projects or in scenarios where transparency, equity, and accountability are non-negotiable requirements.

It has emerged that explainability is not a single, unified concept but varies depending on the end user, thus requiring different approaches based on the system's complexity and its specific applications. We presented an overview of the main explainability techniques, highlighting the differences between symbolic models and machine learning models. Symbolic models, which are inherently interpretable, offer advantages in terms of transparency due to the clarity of their rules and decision-making processes. However, even for symbolic models, advanced techniques such as rule tracing and logical justifications are necessary to facilitate understanding by non-expert users. On the other hand, machine learning models—particularly complex ones such as neural networks—pose significant challenges in terms of interpretability and are therefore often analyzed using post-hoc methods like LIME and SHAP, which aim to make the decisions of opaque models more understandable. Within this context, the explainability framework proposed in this paper stands out for its ability to track and record every single operation through a detailed event-driven system. The combination of a Journal that captures atomic operations, an Explainability Module that translates events into human-readable explanations, and a Resource Storage system ensures complete visibility into the formation and evolution of decisions within the system. This approach guarantees that every automated decision is traceable, understandable, and, most importantly, justifiable. However, the path toward full implementation of explainability remains challenging, as current techniques—while promising—still face limitations related to scalability, the management of sensitive data, and ethical concerns regarding privacy.

In terms of future development, explainability frameworks will require a balanced integration of technical robustness, ethical safeguards, and user-centered design. Future research should address the scalability of explainability solutions (that also in our implementation is still an issue when facing even quasi-big data scale), the protection of sensitive information during explanation generation, and the need for adaptive models that can tailor explanations to diverse user profiles.

Acknowledgment

This research is partially supported by the MAESTRO project, sponsored by the Autonomous Province of Trento – Italy, through L.P. n.6/99 and L.P. n.6/23

References

[1] Gohel P., Singh P. and Mohanty M.. "Explainable AI: current status and future directions." arXiv preprint arXiv:2107.07045 (2021).

[2] Obermeyer Z., Powers B. , Vogeli C. , Mullainathan S., Dissecting racial bias in an algorithm used to manage the health of populations. Science. 2019 Oct 25;366(6464):447-453. doi: 10.1126/science.aax2342. PMID: 31649194.

[3] Arrieta A. et al., "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI." Information fusion 58 (2020): 82-115.

[4] Gomes C. et al. "A Survey of Explainable AI and Proposal for a Discipline of Explanation Engineering." arXiv preprint arXiv:2306.01750 (2023).

[5] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). 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 (pp. 1135–1144). ACM. DOI: 10.1145/2939672.2939778

[6] Lundberg S. M., Lee S., "A unified approach to interpreting model predictions." Advances in neural information processing systems 30 (2017).

[7] Friedman J.H., "Greedy function approximation: a gradient boosting machine." Annals of statistics (2001): 1189-1232.

[8] Apley, D. W., Zhu J., "Visualizing the effects of predictor variables in black box supervised learning models." Journal of the Royal Statistical Society Series B: Statistical Methodology 82.4 (2020): 1059-1086.

[9] Mycin S., A Knowledge-Based Computer Program Applied to Infectious Diseases. Proc Annu Symp Comput Appl Med Care. 1977 Oct 5:66–9. PMCID: PMC2464549.

[10] H. Mercier, D. Sperber, "Why do humans reason? Arguments for an argumentative theory." Behavioral and brain sciences 34.2 (2011): 57-74.

[11] P. Bouquet, A. Molinari and S. Sandri, "Challenges in Working with Unstructured Data in the LLM Era: NER Processes Using Graph Neural Networks," 2024 4th International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), Male, Maldives, 2024, pp. 1-6, doi: 10.1109/ICECCME62383.2024.10796004

[12] Holzinger, A., Langs, G., Denk, H., Zatloukal, K., & Müller, H. (2019). Causability and explainability of artificial intelligence in medicine. Wiley interdisciplinary reviews: data mining and knowledge discovery, 9(4), e1312.

[13] Arya, V., Bellamy, R. K., Chen, P. Y., Dhurandhar, A., Hind, M., Hoffman, S. C., ... & Zhang, Y. (2021, January). Ai explainability 360 toolkit. In Proceedings of the 3rd ACM India joint international conference on data science & management of data (8th ACM IKDD CODS & 26th COMAD) (pp. 376-379).

[14] Reddy, S. (2022). Explainability and artificial intelligence in medicine. The lancet digital health, 4(4), e214-e215.