Semantic Queries Explaining Opaque Machine Learning Classifiers

Jason Liartis[1], Edmund Dervakos[1], Orfeas Menis - Mastromichalakis[1], Alexandros Chortaras[1] and Giorgos Stamou[1]

[1] *Artificial Intelligence and Learning Systems Laboratory, School of Electrical and Computer Engineering, National Technical University of Athens*

Abstract

The success of deep learning on solving a multitude of different problems has highlighted the importance of explainability. In many critical applications, such as in medicine, deep learning cannot be ethically, or lawfully utilized due to its intrinsic opacity. On the other hand, description logics and knowledge representation technologies provide a transparent and intereptetable framework for describing and classifying data. In this paper we present a methodology for utilizing description logics for explaining black-box classifiers. Specifically, given a dataset, a knowledge which describes it and a black-box classifier, we attempt to mimic the black-box with conjunctive queries over the knowledge. These queries are then presented as global explanations of the black-box classifier.

Keywords

XAI, Explainability, Knowledge Graphs

1. Introduction

Over the past decade, Artificial Intelligence and in particular Deep Neural Networks have accomplished impressive achievements in various tasks such as Image Recognition and Natural Language Processing in numerous domains like medicine, finance, arts, and many more. In order to tackle challenging problems, the models become deeper and deeper and their structure complexity is constantly rising, making them inherently opaque. While traditional machine learning models, such as decision trees, are interpretable by design, modern deep neural networks are hard to explain due to their complex architecture and operation. This opacity raises ethical and legal concerns regarding the real-life use of such models and has lead to the emergence of eXplainable Artificial Intelligence (XAI), to make the operation of opaque AI systems more comprehensible to humans [1, 2, 3, 4]. XAI is already a major topic in many events of the AI community and is constantly engaging more researchers due to its immediate and important effect on the application of AI systems.

Knowledge Graphs (KG) and Description Logics (DL) offer an advanced, adaptable and interpretable framework of high expressiveness that can be employed to explain complex neural networks exploiting years of research and improvement on the area. Knowledge graphs [5], as a scalable common understandable structured representation of a domain based on the way humans mentally perceive the world, have emerged as a promising complement or extension to machine learning approaches for achieving explainability [4].

In this paper, we introduce a novel explainability framework for representing model-agnostic knowledge graph-based explanations, as conjunctive queries (resulting to explanations like "animals in images with household items are classified as domestic animals"), approaching the task as a Query Reverse Engineering (QRE) problem, deriving the explanation-queries from the respective outputs of the model. We discuss the creation of appropriate datasets for our methods, and we propose algorithms to compute such explanations. Additionally, we investigate evaluation methods and metrics, and finally, we evaluate the methodology and algorithms experimenting with data from the CLEVR-Hans3 [6] dataset, extracting explanations for deep learning models.

2. Related Work

Approaches to explainability vary with regard to data domain (images, text, tabular), form of explanations (rule-based, counterfactual, feature importance etc.), and scope (global, local) [3]. Closely related to this work are global rule-based explanation methods. These attempt to extract rules based on the predictions of a black-box classifier on a dataset. Many of these methods are based on statistics [7, 8] or extracting rules from decision trees [9, 10], while others are based on logics [11, 12]. Extensionally related to this work are also global prototype explanation methods [13, 14] which present specific representative samples from a dataset as explanations of black-box classifiers.

A key feature of the algorithms proposed in Section 5 is the computation of the Least Common Subsumer (LCS). This problem has been extensively studied for various decription logic expressivities [15, 16, 17, 18]. Furthermore, this work makes use of the strong theoretical and practical results in the area of semantic query answering [19, 20, 21, 22] for evaluating and applying the proposed algorithms in a practical setting.

Numerous works have addressed the Query Reverse Engineering problem, both for knowledge bases and traditional databases [23, 24, 25], and tools have been developed to offer an easy and user friendly way to perform QRE on knowledge bases [26]. The key difference between these works and ours is that we are able to find an approximate solution in cases where there is no exact answer, thanks to the heuristic approach of our algorithms (for more details see Section 5).

3. Background

Let $\mathcal{V} = \langle \mathsf{CN}, \mathsf{RN}, \mathsf{IN} \rangle$ be a *vocabulary*, with $\mathsf{CN}, \mathsf{RN}, \mathsf{IN}$ mutually disjoint finite sets of *concept*, *role* and *individual* names, respectively. Let also $\mathcal{T}$ and $\mathcal{A}$ be a terminology (TBox) and an assertional database (ABox), respectively, over $\mathcal{V}$ using a Description Logics (DL) dialect $\mathcal{L}$, i.e. a set of axioms and assertions that use elements of $\mathcal{V}$ and constructors of $\mathcal{L}$. The pair $\langle \mathcal{V}, \mathcal{L} \rangle$ is a *DL-language*, and $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is a *knowledge base* (KB) over this language. The semantics of KBs are defined in the standard way using interpretations [27]. Given a domain Δ, an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ assigns a set $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ to concept C, a set $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ to role r, and an object $a^{\mathcal{I}} \in \Delta$ to individual a. $\mathcal{I}$ is a *model* of an ABox $\mathcal{A}$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$ for all

$C(a) \in \mathcal{A}$, and $(a^\mathcal{I}, b^\mathcal{I}) \in r^\mathcal{I}$ for all $r(a,b) \in \mathcal{A}$. $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ if it satisfies all axioms in $\mathcal{T}$.

Given a vocabulary $\mathcal{V}$, a *conjunctive query* (simply, a *query*) q is an expression $\{ \langle x_1, \dots x_k \rangle \mid \exists y_1 \dots \exists y_l.(c_1 \wedge \dots \wedge c_n) \}$, where $k, l \geq 0$, $n \geq 1$, x_i, y_i are variable names, each c_i is an atom $C(u)$ or $r(u,v)$, where $C \in \mathsf{CN}$, $r \in \mathsf{RN}$, $u, v \in \mathsf{IN} \cup \{x_i\}_{i=1}^k \cup \{y_i\}_{i=1}^l$ and all x_i, y_i appear in at least one atom. The vector $\langle x_1, \dots x_k \rangle$ is the *head* of q, its elements are the *answer variables*, and $\{c_1, \dots, c_n\}$ is the *body* of q. If q has no answer variables it is *boolean*, and if its body is a singleton, it is *atomic*. In this paper we focus on *non-boolean* queries having *one* answer variable and in which all arguments or all c_i are variables, which are called *instance queries*. For simplicity we write instance queries as $q = \{c_1, ..., c_n\}$, considering always x as the answer variable. Given a KB $\mathcal{K}$, a query q and an interpretation $\mathcal{I}$ of $\mathcal{K}$, an individual $a \in \mathsf{IN}$ is an *answer* to q in $\mathcal{I}$ if it there is a successful variable substitution, substituting $a^\mathcal{I}$ for x. This is formally defined as a *match*, a mapping $\pi : \mathsf{VN}(q) \to \Delta^\mathcal{I}$ such that $\pi(u) \in C^\mathcal{I}$ for all $C(u) \in q$, and $(\pi(u), \pi(v)) \in r^\mathcal{I}$ for all $r(u,v) \in q$. Furthermore, it is a *certain answer* iff it is an answer under all interpretations that are models of $\mathcal{K}$. The set of certain answers to q is $\mathsf{cert}(q, \mathcal{K})$. In some DL dialects, determining the certain answers to a query can be reduced to determining the answers of a special model, called the *canonical model* [28] [29] $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ of $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$. In the dialect we use for our experiments, called Horn $\mathcal{SHIQ}$, this process is even simpler since we can guarantee that the canonical model is finite.

Consider a vocabulary $\mathcal{V} = \langle \mathsf{CN}, \mathsf{RN}, \mathsf{IN} \rangle$, a knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ is the TBox and $\mathcal{A}$ the ABox of $\mathcal{K}$ using a DL dialect $\mathcal{L}$ on $\mathcal{V}$, and the set $\mathcal{Q}$ of all (conjunctive) queries over $\mathcal{V}$. Let $Q \subseteq \mathcal{Q}$ be a set of queries over $\mathcal{K}$. With a slight abuse of notation, we write $\mathsf{cert}(Q, \mathcal{K})$ to denote the set of all answers to the queries of Q, i.e. $\mathsf{cert}(Q, \mathcal{K}) = \cup_{q \in Q} \mathsf{cert}(q, \mathcal{K})$. We can partially order $\mathcal{Q}$ using *query subsumption*: A query q_2 *subsumes* a query q_1 (we write $q_1 \leq_S q_2$) iff there is a substitution θ s.t. $q_2\theta \subseteq q_1$. If q_1, q_2 are mutually subsumed, they are *syntactically equivalent* ($q_1 \equiv_S q_2$).

Let be q an instance query and $q' \subseteq q$. If q' is a minimal subset of q s.t. $q' \leq_S q$, then q' is a *condensation* of q ($\mathsf{cond}(q)$). If that minimal q' is the same set as q, then q is condensed [30].

4. Explaining Opaque ML Classifiers

We are now ready to define and explain the operation of machine learning (ML) classifiers. In this section we only consider boolean classifiers in our definitions, that classify individuals to two classes, but the generalization from boolean to multi-class classifiers is considered trivial.

Definition 1. *Given a set $\mathcal{D} \subseteq \mathsf{IN}$, a* classifier F *defined on $\mathcal{D}$ is a mapping from $\mathcal{D}$ to $\{0,1\}$, i.e.* $\mathsf{F} : \mathcal{D} \to \{0,1\}$. *We call $\mathcal{D}$ the domain of the classifier and with* $\mathsf{F}(\mathcal{D})$ *we denote the set of all elements of $\mathcal{D}$ that are classified to 1, i.e.* $\mathsf{F}(\mathcal{D}) = \{a \in \mathcal{D} \mid \mathsf{F}(a) = 1\}$.

Note that deep neural networks typically classify objects, taking as input other sources of information (images, etc) not included in their semantic description in $\mathcal{K}$. On the other hand, knowledge graphs usually contain this information in the form of datatype properties of objects. Throughout the theoretical analysis we omit this information from $\mathcal{K}$, for simplicity, however for the evaluation we compute $\mathsf{F}(\mathcal{D})$ from images of objects (see Section 6 for details).

Definition 2. *Let $\mathcal{D} \subseteq \mathsf{IN}$ and F be a classifier on $\mathcal{D}$, $\mathcal{K}$ a KB over a vocabulary $\mathcal{V} = \langle \mathsf{CN}, \mathsf{RN}, \mathsf{IN} \rangle$. A query q over $\mathcal{V}$, is a* FO-explanation *(first-order explanation), or simply an* explanation*, of* F *over* $\mathcal{K}$*, iff* $\mathsf{cert}(q, \mathcal{K}) = \mathsf{F}(\mathcal{D})$.

Explanations of ML classifiers do not always exist. Moreover, they are not always unique or helpful in practice. In the following, we introduce the notion of *approximate* explanations , in order to relax the demand to exactly match the output of the classifier. A query q, is an *approximate FO-explanation*, or simply an *approximate explanation* of F, over $\mathcal{K}$, iff $\mathsf{cert}(q, \mathcal{K}) \cap \mathsf{F}(\mathcal{D}) \neq \emptyset$.

Of special interest are approximate explanations that guarantee that all their answers are classified by F as positive. A query q, is an *under-FO-explanation*, or simply an *under-explanation*, of F over $\mathcal{K}$, iff it is an approximate explanation of F and $\mathsf{cert}(q, \mathcal{K}) \subseteq \mathsf{F}(\mathcal{D})$.

Of interest are also approximate explanations for which all individuals classified by F as positive are in their answer set. A query q, is an *over-FO-explanation*, or simply an *over-explanation*, of F over $\mathcal{K}$, iff it is an approximate explanation of F and $\mathsf{F}(\mathcal{D}) \subseteq \mathsf{cert}(q, \mathcal{K})$. In most cases multiple approximate explanations exist, so we need to find a way to evaluate them. In the following paragraph we propose a set of evaluation metrics for such explanations.

Explanation Evaluation Metrics We can define similarity measures that take into account the syntactical form of the queries, their certain answers over any ABox, or their certain answers over a specific knowledge base. For the syntactic similarity of two queries, there is a lot of work in the area (see for example the work on query refinement and graph distance). For answer-based similarity for any ABox, it is not obvious how to define similarity measures, since it is difficult to consider all ABoxes or to capture all the TBox information by checking the syntax. The case of answer-based similarity for a specific knowledge base is more intuitive and can be addressed using either the well-known Jaccard similarity coefficient defined on sets, or other popular metrics from the area of machine learning like precision and recall. So, we define the following three similarity measures on $\mathcal{Q}$ for a query-explanation q of a classifier F over a specific knowledge base $\mathcal{K}$ and a set $\mathcal{D} \subseteq \mathsf{IN}$ as follows:

Degree A Jaccard distance-based similarity measure, it's obvious that the degree of an (exact) explanation is equal to 1. $\mathsf{deg}(q, \mathsf{F}) = \frac{|\mathsf{cert}(q,\mathcal{K}) \cap \mathsf{F}(\mathcal{D})|}{|\mathsf{cert}(q,\mathcal{K}) \cup \mathsf{F}(\mathcal{D})|}$

Precision A similarity measure inspired by the precision of ML classifiers, it's obvious that the precision of an under-explanation is equal to 1. $\mathsf{pre}(q, \mathsf{F}) = \frac{|\mathsf{cert}(q,\mathcal{K}) \cap \mathsf{F}(\mathcal{D})|}{|\mathsf{cert}(q,\mathcal{K})|}$

Recall A similarity measure inspired by the recall of ML classifiers, it's obvious that the recall of an over-explanation is equal to 1. $\mathsf{rec}(q, \mathsf{F}) = \frac{|\mathsf{cert}(q,\mathcal{K}) \cap \mathsf{F}(\mathcal{D})|}{|\mathsf{F}(\mathcal{D})|}$

5. Computing Explanations

In this section we describe in detail the proposed algorithms, and introduce the concept of an *Explanation Dataset*, which plays a key role for effectively utilizing the proposed algorithms in a practical setting.

5.1. Algorithms

In order to unify explanations we use the notion of the *least common subsumer* (LCS). Given two queries q_1, q_2, their least common subsumer $\mathsf{LCS}(q_1, q_2)$ is defined as the query q for which $q_1, q_2 \leq_S q$ and $\forall q' : q_1,\ q_2 \leq_S q' \Leftrightarrow q \leq_S q'$. The least common subsumer is the most specific generalization of q_1, q_2.

In order to manipulate queries and compute explanations, we use the representation of interpretations and queries as labeled graphs. For interpretations, each element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ is represented by a node with a label which contains every concept $C \in \mathsf{CN}$ for which $a^{\mathcal{I}} \in C^{\mathcal{I}}$. Two nodes $a^{\mathcal{I}}, b^{\mathcal{I}}$ are connected with an edge with label r if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$. This graph can be described by a triple (V, E, L) where $V = \Delta^{\mathcal{I}}$ are the nodes $a^{\mathcal{I}}$, $E \subseteq \Delta^{\mathcal{I}} \times \mathsf{RN} \times \Delta^{\mathcal{I}}$ are the edges $(a^{\mathcal{I}}, r, b^{\mathcal{I}})$, and $L : V \to 2^{\mathsf{CN}}$ is the labeling function $L(a^{\mathcal{I}}) = \{C, D, \dots\}$ of the nodes. The graph representation for queries is defined in the same way, but with nodes corresponding to variables and labels and edges to the conjuncts containing those variables.

These representations are useful because a lot of the concepts we have presented so far can be rephrased in terms of homomorphisms between labeled graphs. Given two labeled graphs $G_1 = (V_1, E_1, L_1)$, $G_2 = (V_2, E_2, L_2)$, a mapping $h : V_1 \to V_2$ is called a homomorphism iff it respects the structure of G_1, or more formally: (i) $\forall v \in V_1,\ L_1(v) \subseteq L_2(h(v))$ and (ii) $\forall u, v \in V_1,\ (u, r, v) \in E_1 \Rightarrow (h(u), r, h(v)) \in E_2$. If such a mapping exists we say that G_1 is homomorphic to G_2 and we write $G_1 \to G_2$.

A match π can now be rephrased as a homomorphism from the query graph of q to the interpretation graph of I, and a is an answer to q if there is such a homomorphism with $\pi(x) = a^{\mathcal{I}}$ and a certain answer if there is a homomorphism to the canonical model with $\pi(x) = a^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$. Subsumption can also be described in terms of homomorphisms since its definition implies that $G_{q_1} \to G_{q_2} \Leftrightarrow q_2 \leq_S q_1$, where G_q is the graph of query q.

In order to calculate the LCS we use an extension of the Kronecker product of graphs to labeled graphs. Given two labeled graphs $G_1 = (V_1, E_1, L_1)$, $G_2 = (V_2, E_2, L_2)$ the Kronecker product $G = G_1 \times G_2$ of those graphs is: $G = (V, E, L),\ V = V_1 \times V_2$ (Cartesian product of sets), $((u_1, r, v_1) \in E_1,\ (u_2, r, v_2) \in E_2) \Leftrightarrow ((u_1, u_2), r, (v_1, v_2)) \in E,\ L((v_1, v_2)) = L_1(v_1) \cap L_2(v_2)$

As with the Kronecker product of unlabeled graphs, it holds that $H \to G_1, G_2 \Leftrightarrow H \to G_1 \times G_2$, this implies that the graph of $\mathsf{LCS}(q_1, q_2)$ is $G_{q_1} \times G_{q_2}$, with the node (x, x) becoming the new answer variable and the other nodes of $G_{q_1} \times G_{q_2}$ are renamed arbitrarily. Calculating the LCS using the Kronecker product involves $O(n^2 m^2)$ operations, where $n = |V_1|$, $m = |V_2|$.

Even though the Kronecker product between two queries computes the LCS, the query it produces is not minimal with respect to the number of variables. Since these queries are intended to be shown to humans as explanations, the minimization of the number of variables is imperative. However, condensing a query is coNP-complete [30]. For this reason, we utilize Algorithm 1 which removes redundant conjuncts and variables, however without a guarantee of producing a fully condensed query.

For each pair of variables present in a query, Algorithm 1 checks if unifying the two variables is equivalent to deleting one of the two, in which case the variable and conjuncts are deleted. Verifying the correctness of Algorithm 1 amounts to verifying the claim of line 6. By unifying variable v_j with v_i, all conjuncts of the form $C(v_j)$ become $C(v_i)$ and all conjuncts of the forms

$r(v_j, v_k), r(v_k, v_j), r(v_j, v_j)$ become respectively $r(v_i, v_k), r(v_k, v_i), r(v_i, v_i)$. Line 6 checks that those conjuncts are already present in the query and therefore that unifying v_j with v_i is equivalent to deleting v_j. Unifying two variables of q produces a query that is subsumed by q, while deleting a variable produces a query that subsumes q, therefore Algorithm 1 produces syntactically equivalent queries. For the complexity of 1 refer to the appendix.

Algorithm 1: MINIMIZE

```
   Input: Query $q = (V, E, L)$ to be minimized.
   Output: The minimized query $q'$.
1  $n \leftarrow V$
2  $q' \leftarrow q$
3  do
4      $q \leftarrow q'$
5      foreach pair $0 < i, j \leq n$, $i \neq j$ do
6          Check if unifying $v_j$ of $q'$ with $v_i$ of $q'$ would be the same as deleting it:
7          if $L(v_j) \subseteq L(v_i)$ and $((v_j, r, v_k) \in E \Rightarrow (v_i, r, v_k) \in E,\ k \neq j)$ and $((v_k, r, v_j) \in E \Rightarrow (v_k, r, v_i) \in E,\ k \neq j)$ and $((v_j, r, v_j) \in E \Rightarrow (v_i, r, v_i) \in E)$ then
8              Delete variable $j$ from $q'$.
9          end
10     end
11 while $q' \neq q$
12 return $q'$
```

We can also use the graph representation of the canonical model to introduce the notion of the *most specific query* (MSQ) of an individual a. That is the query that contains as much information as possible about a, it is the least query in terms of subsumption that has that a as a certain answer and it can be constructed by converting the connected component of the graph of $\mathcal{C}_{\mathcal{T},\mathcal{A}}$ that contains a into a query graph with the node of $a^{\mathcal{C}_{\mathcal{T},\mathcal{A}}}$ becoming the answer variable. Every query that has a as a certain answer must be homomorphic to that connected component and therefore to the MSQ as well.

For generating explanations of black-box classifiers, we propose Algorithm 2, which generates candidate approximate explanations for sets of individuals, by utilizing a heuristic for disjointness which is discussed later in this section. For generating explanations, Algorithm 2 first populates a list with the MSQ of each individual. Then it removes the two least disjoint (according to the heuristic) queries and replaces them with their LCS (computed as the Kronecker product) only if it has fewer variables than a pre-set threshold, after it is minimized with Algorithm 1. The LCS of the two least disjoint queries is also added to the set of candidate explanations to be returned. This process is repeated until there are fewer than two queries left to check for disjointness.

For heuristically approximating disjointness between two queries we use Algorithm 3. Algorithm 3 computes a rough estimate of how disjoint two queries are. It compares every variable v_1 of q_1 with every variable v_2 of q_2 counting how many concept and role conjuncts containing v_1 would certainly be removed if v_1 was unified with v_2. It keeps the best such count for v_1 and

Algorithm 2: EXPLAIN

```
Input: A set of individuals $\{i_1, i_2, \dots i_n\} \subseteq \mathsf{IN}$ and a threshold $t$ of maximum query size.
Output: A set of queries as explanations of the individuals.
explanations $\leftarrow \varnothing$
queries $\leftarrow \{\text{msq}(i_j)\}_{j=1}^{n}$
while 'queries' has two or more elements do
    Find the least disjoint pair of queries:
    $q_1, q_2 \leftarrow \arg\min\{\text{disj}(q, q') \mid q, q' \in \text{queries}\}$
    Remove $q_1, q_2$ from 'queries'.
    $q \leftarrow \text{minimize}(\mathsf{LCS}(q_1, q_2))$
    if the number of variables in $q$ is $\leq t$ then
        explanations $\leftarrow$ explanations $\cup \{q\}$
        queries $\leftarrow$ queries $\cup \{q\}$
    end
end
return explanations
```

adds it to the estimate of the disjointness. This process is then symmetrically repeated for the variables of q_2.

5.2. Explanation Dataset

The connecting component between our work and machine learning classifiers which drives this explanation framework is the explanation dataset. It is essentially a semantically enriched dataset compatible with the classifier under investigation. It consists of data that can be fed to the classifier (e.g. images for image classifiers) along with semantic descriptions of this data expressed with description logics. The additional information of this enriched dataset allows us to produce explanations for the classifiers exploiting the algorithms mentioned in Section 5.1.

There isn't a standard procedure for the semantic enrichment of data, so we need to find a way to create these explanation datasets. Thankfully, enriched datasets already exist within the premises of machine learning like the Visual Genome [31] or the CLEVR [32] datasets, which contain images along with metadata for each image, such as annotations or sets of questions-answers. It is usually easy to express these metadata in a description logic, in most cases mapping them to individuals, concepts, roles and axioms. Such an example is the creation of an explanation dataset from the CLEVR-Hans3 dataset [6] shown in 6.1.

6. Experiments and Discussion

6.1. Explanation Dataset Creation

CLEVR-Hans3 [6] is a confounded image classification dataset, designed to evaluate algorithms that detect and fix biases of classifiers. It consists of CLEVR [32] images divided into three

Algorithm 3: DISJ

```
Input: A pair of queries $q_1 = (V_1, E_1, L_1)$, $q_2 = (V_2, E_2, L_2)$ for which to calculate a
        very rough estimate of how disjoint they are.
Output: The estimate of the disjointness.
disj $\leftarrow 0$
For every variable in $q_1$ find how much it differs in terms of labels and number of edges
  from its closest match in $q_2$:
foreach $v_1 \in V_1$ do
    min_diff $\leftarrow +\infty$
    foreach $v_2 \in V_2$ do
        diff $\leftarrow |L_1(v_1) \setminus L_2(v_2)|$
        for $r \in \mathsf{RN}$ do
            diff $\leftarrow$ diff
             $+ \max\{|\{(v_1, r, u_1) \in E_1, u_1 \in V_1\}| - |\{(v_2, r, u_2) \in E_2, u_2 \in V_2\}|, 0\}$
            diff $\leftarrow$ diff
             $+ \max\{|\{(u_1, r, v_1) \in E_1, u_1 \in V_1\}| - |\{(u_2, r, v_2) \in E_2, u_2 \in V_2\}|, 0\}$
        end
        min_diff $\leftarrow$ min(min_diff, diff)
    end
    disj $\leftarrow$ disj + min_diff
end
Repeat the above but with $q_1$ and $q_2$ reversed.
return disj
```

classes, of which two are confounded. The membership of a class is based on combinations of objects' attributes and relations. Thus, within the dataset, consisting of train, validation, and test splits, all train, and validation images of confounded classes will be confounded with a specific attribute. The rules that the classes follow are the following, with the confounding factors in parentheses: (i) Large (Gray) Cube and Large Cylinder, (ii) Small Metal Cube and Small (Metal) Sphere, (iii) Large Blue Sphere and Small Yellow Sphere.

We created our explanation dataset using the test set of CLEVR-Hans3, consisting of 750 images for each class. Exploiting the description of the images provided in json files, we constructed a vocabulary $\langle \mathsf{CN}, \mathsf{RN}, \mathsf{IN} \rangle$, with all the images and the objects therein as individuals (IN), the concepts defining the size, color, shape, and material of each object, as well as two indicative concepts Image and Object as concept names (CN), and the role "contains(Image, Object)" indicating the existence of an object in a specific image as the only role name (RN). We then created a knowledge base over this vocabulary, with the ABox containing the semantic description of all images and the respective objects, and the TBox containing certain rather trivial inclusion axioms. The sets CN, RN and the Tbox of our knowledge base and the respective vocabulary are the following: $\mathsf{CN} = \{$Image, Object, Cube, Cylinder, Sphere, Metal, Rubber, Blue,Brown, Cyan, Gray, Green, Purple, Red, Yellow, Large, Small$\}$, $\mathsf{RN} = \{$contains(Image, Object)$\}$, $\mathcal{T} = \{\mathsf{x} \sqsubseteq \mathsf{Object}\}$ (where $x \notin \{\mathsf{Image}, \mathsf{Object}\}$).

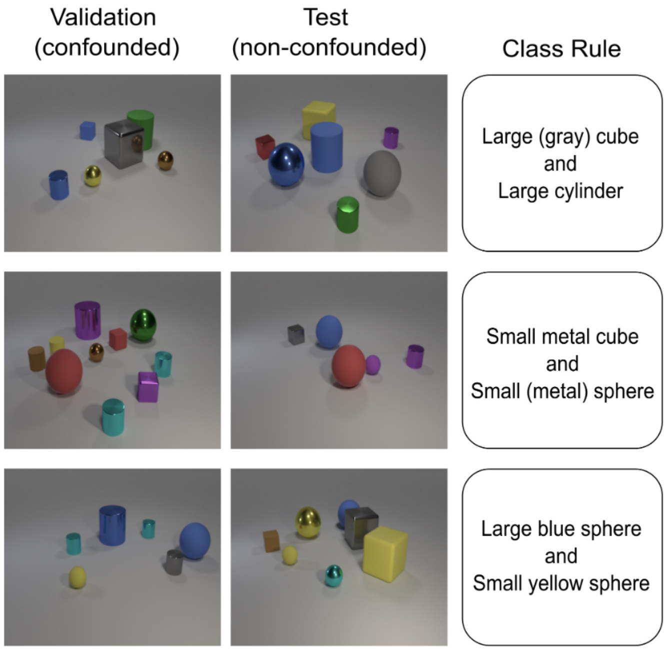
Validation (confounded)
Test (non-confounded)
Class Rule
Large (gray) cube and Large cylinder
Small metal cube and Small (metal) sphere
Large blue sphere and Small yellow sphere

Figure 1: The three classes of CLEVR-Hans3 with their rules and the confounding factors in parentheses.

6.2. Setting

For CLEVR-Hans3 we used the same classifier and training procedure as the one used by the creators of the dataset in [33]. The classifier is a deep CNN, specifically ResNet34 [34]. The performance of the classifier is shown in Table 1. After training the classifier we acquired its predictions on the test set and generated explanations for each class with Algorithm 2. We also loaded the explanation dataset in GraphDB [1] for getting certain answers of queries, and evaluating explanations.

Table 1
Performance of ResNet34 on CLEVR-Hans3.

	Test set metrics			Confusion martix		
True label	Precision	Recall	F1-score	Class 1	Class 2	Class 3
Class 1	0.94	0.16	0.27	118	511	121
Class 2	0.59	0.98	0.54	5	736	9
Class 3	0.85	1.00	0.92	2	0	748

[1] https://graphdb.ontotext.com/

6.3. Results

The explanations generated for the classifier on the test set of CLEVR-Hans3 are shown in Table 2, where we show the query, the value of each metric and the numbers of positive and negative individuals. The term positive individuals refers to the certain answers of the query that are classified to the respective class, while the term negative individuals refers to the rest of the certain answers. In our representation of the queries in Table 2 we have omitted the answer variable x, along with all conjuncts of the form x contains y and conjuncts of the form Object(y), for brevity. The algorithm found an under-explanation (precision=1) and an over-explanation (recall=1) for each class, with the best explanation degree achieved for class 3, which lacks a confounding factor. Over-explanations and under-explanations are of particular interest since they can be translated into global rules which the classifier follows on the particular dataset. For instance the under-explanation for class 1 is translated into the rule "If the image contains a Large Gray Cube, a Large Cylinder and a Large Metal Object then it is classified to class 1.", while the over-explanation for the same class is translated into the rule "If the image does not contain a Large Cube then it is not classified to class 1". We can see that over-explanations tend to be more general, in order to include all the predictions of the classifier, while on the other side, under-explanations tend to be much more specific. That's why we find approximate explanations to be very useful for the general description of a classifier. Both over- and under-explanations tend to have low degree, while in general explanations with high degree provide us with a more accurate approximation of what the classifier has learned. So it is also useful to consider approximate explanations of high degree, even though they cannot be translated to rules like over- and under-explanations.

It is interesting to note that the over-explanation produced for class 1 contains a Large Cube but not a Large Cylinder. This gives us the information that in the training process the classifier learned to pay more attention to the presence of cubes rather than the presence of cylinders. The elements of the under-explanation that differ from the true rule of class 1 can be a great starting point for a closer inspection of the classifier. We expected the presence of a Gray Cube from the confounding factor introduced in the training and validation sets, but in a real world scenario similar insights can be reached by inspecting the queries. In our case, we further inquired the role that the Gray Cube and the Large Metal Object play in the under-explanation by removing either of them from the query and examining its new performance. In Table 3 we can see that the gray color was essential for the under-explanation while the Large Metal Object was not, and in fact its removal improved the under-explanation and returned almost the entire class.

Another result that piqued our attention is the approximate explanation for class 3 which is the actual rule that describes this class. This explanation returns two negative individuals which we can also see in the confusion matrix of the classifier and we were interested to examine what sets these two individuals apart. We found that both of these individuals are answers to the query "y1 is Large, Gray, Cube". This shows us once again the large effect the confounding factor of class 1 had on the classifier.

Our overall results show that the classifier tends to emphasize low level information such as color and shape and ignores high level information such as texture and the combined presence of multiple objects. This is the reason why the confounding factor of class 1 had an important effect to the way images are classified, while the confounding factor of class 2 seems to have had

Table 2
Optimal explanations with regard to the three metrics on CLEVR-Hans3.

Metric	Query	Precision	Recall	Degree	Positives
		Class 1			
Best Precision	y1 is Large, Cube, Gray. y2 is Large, Cylinder. y3 is Large, Metal.	1.00	0.66	0.66	83
Best Recall	y1 is Large, Cube.	0.09	1.00	0.09	125
Best Degree	y1 is Large, Cube, Gray. y2 is Large, Cylinder. y3 is Large, Metal.	1.00	0.66	0.66	83
		Class 2			
Best Precision	y1 is Small, Sphere. y2 is Large, Rubber. y3 is Small, Metal, Cube. y4 is Small, Brown. y5 is Small, Rubber, Cylinder.	1.00	0.09	0.09	116
Best Recall	y1 is Cube.	0.63	1.00	0.63	1247
Best Degree	y1 is Metal, Cube. y2 is Small, Metal.	0.78	0.8	0.65	1005
		Class 3			
Best Precision	y1 is Metal, Blue. y2 is Large, Blue, Sphere. y3 is Yellow, Small, Sphere. y4 is Small, Rubber. y5 is Metal, Sphere.	1.00	0.42	0.42	365
Best Recall	y1 is Large. y2 is Sphere.	0.42	1.00	0.42	878
Best Degree	y1 is Yellow, Small, Sphere. y2 is Large, Blue, Sphere.	0.99	0.85	0.85	748

a much smaller one.

7. Conclusion

We introduced a theoretical framework for representing global explanations for ML classifiers in the form of conjunctive queries. We also developed algorithms for computing explanations for limited knowledge bases, and investigated some of their quality-related properties. Using

Table 3
Two modified versions of the class 1 under-explanation produced by removing conjuncts.

Query	Positives	Negatives
y1 is Large, Cube. y2 is Large, Cylinder. y3 is Large, Metal.	108	547
y1 is Large, Cube, Gray. y2 is Large, Cylinder.	93	0

CLEVR-Hans3, we were able to generate multiple explanations for a deep learning classifier. In several cases, we found the explanations to be useful for detecting potential biases, and providing a more intuitive approach for evaluating classifiers besides performance metrics which are typically used.

One of the conclusions we can draw from our experiments is the importance of developing methods evaluating explanations, which we plan to explore in future work. The three metrics used in this paper are simple and intuitive, however none take under consideration the human-readability factor which is crucial for explainability.

Finally, the quality and usefulness of explanations generated with the proposed methodology depends on the characteristics of the *explanation dataset*. Thus, in future work we also plan to investigate what constitutes a “good" explanation dataset, in addition to experimenting on more specialized domains in which explainability is critical, such as in medical applications.

References

[1] M. Turek, Explainable artificial intelligence (xai), Defense Advanced Research Projects Agency. http://web. archive. org/web/20190728055815/https://www. darpa. mil/program/explainable-artificial-intelligence (2018).

[2] W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, B. Yu, Interpretable machine learning: definitions, methods, and applications, CoRR abs/1901.04592 (2019).

[3] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, D. Pedreschi, A survey of methods for explaining black box models, ACM Comput. Surv. 51 (2019) 93:1–93:42.

[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, F. Herrera, Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI, Inf. Fusion 58 (2020) 82–115.

[5] A. Hogan, E. Blomqvist, M. Cochez, C. d’Amato, G. de Melo, C. Gutiérrez, J. E. L. Gayo, S. Kirrane, S. Neumaier, A. Polleres, R. Navigli, A. N. Ngomo, S. M. Rashid, A. Rula, L. Schmelzeisen, J. F. Sequeda, S. Staab, A. Zimmermann, Knowledge graphs, CoRR abs/2003.02320 (2020).

[6] W. Stammer, P. Schramowski, K. Kersting, Right for the right concept: Revising neuro-symbolic concepts by interacting with their explanations, arXiv preprint arXiv:2011.12854 (2020).

[7] H. Yang, C. Rudin, M. I. Seltzer, Scalable bayesian rule lists, in: Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, 2017, pp. 3921–3930. URL: http://proceedings.mlr.press/v70/yang17h.html.

[8] Y. Ming, H. Qu, E. Bertini, Rulematrix: Visualizing and understanding classifiers with rules, IEEE Trans. Vis. Comput. Graph. 25 (2019) 342–352. URL: https://doi.org/10.1109/TVCG.2018.2864812. doi:10.1109/TVCG.2018.2864812.

[9] M. W. Craven, J. W. Shavlik, Extracting tree-structured representations of trained networks, in: Advances in Neural Information Processing Systems 8, NIPS, Denver, CO, USA, November 27-30, 1995, 1995, pp. 24–30. URL: http://papers.nips.cc/paper/1152-extracting-tree-structured-representations-of-trained-networks.

[10] Y. Zhou, G. Hooker, Interpreting models via single tree approximation, arXiv preprint arXiv:1610.09036 (2016).

[11] M. K. Sarker, N. Xie, D. Doran, M. Raymer, P. Hitzler, Explaining trained neural networks with semantic web technologies: First steps, in: NeSy, volume 2003 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2017.

[12] G. Ciravegna, F. Giannini, M. Gori, M. Maggini, S. Melacci, Human-driven FOL explanations of deep learning, in: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, 2020, pp. 2234–2240. URL: https://doi.org/10.24963/ijcai.2020/309. doi:10.24963/ijcai.2020/309.

[13] B. Kim, O. Koyejo, R. Khanna, Examples are not enough, learn to criticize! criticism for interpretability, in: Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain, 2016, pp. 2280–2288. URL: https://proceedings.neurips.cc/paper/2016/hash/5680522b8e2bb01943234bce7bf84534-Abstract.html.

[14] K. S. Gurumoorthy, A. Dhurandhar, G. A. Cecchi, C. C. Aggarwal, Efficient data representation by selecting prototypes with importance weights, in: 2019 IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8-11, 2019, 2019, pp. 260–269. URL: https://doi.org/10.1109/ICDM.2019.00036. doi:10.1109/ICDM.2019.00036.

[15] W. W. Cohen, A. Borgida, H. Hirsh, Computing least common subsumers in description logics, in: Proceedings of the 10th National Conference on Artificial Intelligence, San Jose, CA, USA, July 12-16, 1992, 1992, pp. 754–760. URL: http://www.aaai.org/Library/AAAI/1992/aaai92-117.php.

[16] R. Küsters, R. Molitor, Structural subsumption and least common subsumers in a description logic with existential and number restrictions, Stud Logica 81 (2005) 227–259. URL: https://doi.org/10.1007/s11225-005-3705-5. doi:10.1007/s11225-005-3705-5.

[17] F. Baader, B. Sertkaya, A. Turhan, Computing the least common subsumer w.r.t. a background terminology, J. Appl. Log. 5 (2007) 392–420. URL: https://doi.org/10.1016/j.jal.2006.03.002. doi:10.1016/j.jal.2006.03.002.

[18] F. M. Donini, S. Colucci, T. D. Noia, E. D. Sciascio, A tableaux-based method for computing least common subsumers for expressive description logics, in: Proceedings of the 22nd International Workshop on Description Logics (DL 2009), Oxford, UK, July 27-30, 2009, 2009. URL: http://ceur-ws.org/Vol-477/paper_22.pdf.

[19] D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, R. Rosati, Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family, J. Autom. Reason. 39 (2007) 385–429.

[20] B. C. Grau, B. Motik, G. Stoilos, I. Horrocks, Computing datalog rewritings beyond horn

ontologies, in: IJCAI, IJCAI/AAAI, 2013, pp. 832–838.

[21] A. Chortaras, M. Giazitzoglou, G. Stamou, Inside the query space of DL knowledge bases, in: Description Logics, volume 2373 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2019.

[22] D. Trivela, G. Stoilos, A. Chortaras, G. Stamou, Resolution-based rewriting for horn-*SHIQ* ontologies, Knowl. Inf. Syst. 62 (2020) 107–143.

[23] M. Arenas, G. I. Diaz, E. V. Kostylev, Reverse engineering SPARQL queries, in: J. Bourdeau, J. Hendler, R. Nkambou, I. Horrocks, B. Y. Zhao (Eds.), Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016, ACM, 2016, pp. 239–249. URL: https://doi.org/10.1145/2872427.2882989. doi:10.1145/2872427.2882989.

[24] Q. T. Tran, C. Y. Chan, S. Parthasarathy, Query reverse engineering, VLDB J. 23 (2014) 721–746. URL: https://doi.org/10.1007/s00778-013-0349-3. doi:10.1007/s00778-013-0349-3.

[25] A. Petrova, E. V. Kostylev, B. C. Grau, I. Horrocks, Query-based entity comparison in knowledge graphs revisited, in: C. Ghidini, O. Hartig, M. Maleshkova, V. Svátek, I. F. Cruz, A. Hogan, J. Song, M. Lefrançois, F. Gandon (Eds.), The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part I, volume 11778 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 558–575. URL: https://doi.org/10.1007/978-3-030-30793-6_32. doi:10.1007/978-3-030-30793-6_32.

[26] G. Diaz, M. Arenas, M. Benedikt, Sparqlbye: Querying rdf data by example, Proc. VLDB Endow. 9 (2016) 1533–1536. URL: https://doi.org/10.14778/3007263.3007302. doi:10.14778/3007263.3007302.

[27] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation, and Applications, Cambridge University Press, 2003.

[28] R. Kontchakov, M. Zakharyaschev, An Introduction to Description Logics and Query Rewriting, Springer International Publishing, Cham, 2014, pp. 195–244. URL: https://doi.org/10.1007/978-3-319-10587-1_5. doi:10.1007/978-3-319-10587-1_5.

[29] F. Baader, I. Horrocks, C. Lutz, U. Sattler, An Introduction to Description Logic, Cambridge University Press, 2017. doi:10.1017/9781139025355.

[30] G. Gottlob, C. G. Fermüller, Removing redundancy from a clause, Artif. Intell. 61 (1993) 263–289.

[31] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, et al., Visual genome: Connecting language and vision using crowdsourced dense image annotations, arXiv preprint arXiv:1602.07332 (2016).

[32] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, R. B. Girshick, CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning, CoRR abs/1612.06890 (2016). URL: http://arxiv.org/abs/1612.06890. arXiv:1612.06890.

[33] W. Stammer, P. Schramowski, K. Kersting, Right for the right concept: Revising neuro-symbolic concepts by interacting with their explanations, CoRR abs/2011.12854 (2020).

[34] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, CoRR

abs/1512.03385 (2015). URL: http://arxiv.org/abs/1512.03385. arXiv:1512.03385.

A. Complexity of Algorithms

A.1. Minimize

Given n as the number of variables present in a query, then loop 3 will be executed at most n times, since at each loop either a variable is deleted or the algorithm returns. Loop 5 checks all pairs of variables ($O(n^2)$), and condition 7 requires $O(n)$ set comparisons and $2|\mathsf{RN}|$ comparisons of rows and columns of adjacency matrices. Thus the complexity of Algorithm 1 is $O(n^4)$.

A.2. Disj

If the number of variables present in each query is n and m, then the time complexity of the algorithm is $O(n \cdot m)$ due to the two outer loops, while the inner loop and other computations require constant time with some careful pre-processing of the queries so that the edge count is readily available.

A.3. Explain

Regarding the complexity of Algorithm 2, having computed the complexity of its components, let n be the number of individuals, m be the maximum number of variables present in an individual's MSQ and t be the threshold for the number of variables, above which we discard the resulting queries. Finding the connected component which corresponds to each individual's MSQ requires $O(m^2)$ operations, thus initializing the *queries* list can be done in $O(nm^2)$. Then, at each loop, the algorithm removes two queries from the list, and depending on if the condition is satisfied, it adds a query to the list, thus the loop will be executed at most $n-1$ times. If we have stored the values of disj(q, q') after computing them for the first time, then we can find the pair of queries which minimize it with $O(n^2)$ operations, while computing them for the first time has a time complexity of $O(m^2)$ for each pair of queries as shown in the previous paragraph. Initially there are $\frac{n(n-1)}{2}$ pairs, while on iteration i, by adding a new query to the list, $n-i$ new pairs are created. The maximum number of variables for created queries is $t \geq m$, thus all executions of finding the least disjoint pair of queries will require $O(n^3 + n^2t^2)$ operations. Finally computing the LCS can be done in $O(t^4)$, while the complexity of minimization is also $O(t^4)$, resulting in the time complexity of Algorithm 2 as $O(n^3 + n^2t^2 + nt^4)$.