Spatial Relation Learning for Explainable Image Classification and Annotation in Critical Applications

Régis Pierrard[1,2], Jean-Philippe Poli[1], and Céline Hudelot[2]

[1]CEA, LIST, 91191 Gif-sur-Yvette cedex, France.
[2]Paris-Saclay University, CentraleSupélec, Mathematics Interacting with Computer Science, 91190, Gif-sur-Yvette, France.
{*regis.pierrard, jean-philippe.poli*}*@cea.fr, celine.hudelot@centralesupelec.fr*

Abstract

With the recent successes of black-box models in Artificial Intelligence (AI) and the growing interactions between humans and AIs, explainability issues have risen. In this article, in the context of high-stake applications, we propose an approach for explainable classification and annotation of images. It is based on a transparent model, whose reasoning is accessible and human understandable, and on interpretable fuzzy relations that enable to express the vagueness of natural language. The knowledge about relations is set beforehand by an expert and thus training instances do not need to be annotated. The most relevant relations are extracted using a fuzzy frequent itemset mining algorithm in order to build rules, for classification, and constraints, for annotation. We also present two heuristics that make the process of evaluating relations faster. Since the strengths of our approach are the transparency of the model and the interpretability of the relations, an explanation in natural language can be generated. Supported by experimental results, we show that, given a segmentation of the input, our approach is able to successfully perform the target task and generate explanations that were judged as consistent and convincing by a set of participants.

1 Introduction

Explainable Artificial Intelligence (XAI) has recently received special attention. XAI aims at obtaining a relative understanding of what a model does or why a decision was made [57]. It involves several different fields such as machine learning, knowledge representation and reasoning, cognitive science or law and provides a set of tools that enable to build explainable models, to extract post-hoc interpretations from black-box models or to evaluate explanations.

An explanation involves an explainer, the model, and an explainee, the end-user. Consequently, as illustrated in Figure 1, the type of explanations, and thus the XAI method we rely on, should depend on:

- The *application*: different applications may require different models and so different approaches for explaining them. For example, we can resort to a post-hoc interpretability method for interpreting a deep neural network while a decision tree is intrinsically explainable (assuming that it is shallow enough to be considered explainable).
- The *end-user*: the form of the explanation should be strongly dependent on the knowledge of the end-user. First, if this user has no knowledge about the model being used, high-level explanations may be required. Second, the vocabulary used to build the explanations should be suited to the user.

We focus here on high-stake applications of AI, where the cost of making a mistake is high. For such applications, it has been claimed that black-box models are not appropriate because getting a faithful and detailed enough explanation is tricky [70]. Indeed, in most cases, it consists in attributing importance scores to features [69, 53, 72] that are not necessarily interpretable, like pixels in an image [69, 3]. In such critical applications, *transparent models*, i.e. models whose trace and reasoning can be conveniently tracked, are better suited for producing reliable explanations. However, such models are not suited for dealing with images because explanations should not be based on raw input features (pixels) but on

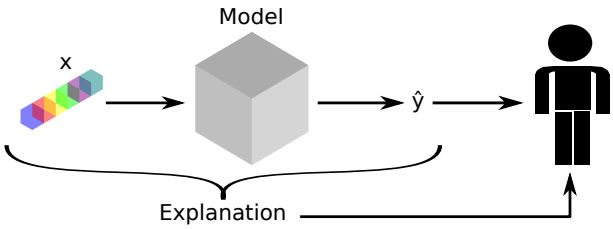


Figure 1: An explanation is provided to a user and should render the reasoning of the model under study on the features (denoted as x) it was fed with.

features that are semantically interpretable. Understanding an image relies on *understanding the relations* between the entities in this image [8, 26], and thus these relations should be part of the explanations.

In this article, we propose an approach that aims at building an explainable model for performing classification or annotation of images. Relying on a human-in-the-loop strategy [33], a vocabulary of potentially relevant relations is set beforehand. Those are *fuzzy* relations, which enable to manage the imprecision of the data and the vagueness of the language used for generating the model and the explanations. Then, the most frequent sets of relations are extracted to build rules for classification or constraints for defining a fuzzy constraint satisfaction problem in the case of annotation. Since our model is fully transparent and our relations are interpretable, an explanation in natural language can be generated for each decision in order to achieve causability [34]. While learning fuzzy relations to build rules for classification was proposed by Gonzalez *et al.* [28], their approach was not able to deal with images and annotation problems.

This article is structured as follows. Section 2 gives a brief overview of the types of transparent models that were proposed in the literature. This enables us to specify how the model we want to build differs from them. We also remind how the interpretability of state-of-the-art approaches is evaluated when dealing with images. In Section 3, we remind the core theoretical notions that our approach relies on. The following section presents the whole approach for building an explainable classifier or annotator. Then, we describe two heuristics that enable to evaluate relations in an efficient way, which is paramount since a brute force approach is too time-consuming. Section 6 is dedicated to presenting the experiments we carried out to assess our model on classification and annotation tasks. Finally, we discuss our results and the prospects we envision.

2 Related Works

In this section, we present the two types of approaches that can be performed to generate explanations from a model working on images. First, we talk about models that are inherently transparent and thus explainable by design. The other type of approaches consists in using a black-box model and interpreting it afterwards with a post-hoc method.

2.1 Explainable Models by Design

While there has been recently a surge of interest for explaining the decisions returned by AIs, this is not a new problem. The first expert system, Dendral [22, 52], was released in 1965 for describing the molecular structure of unknown organic chemical compounds. It was not able to explain its outputs but it could provide a trace of the program reasoning steps. The first intelligent systems that were able to provide an explanation were developed in the 1970's. Among them, MYCIN was one of the first rule-based expert systems [71]. It was an automated consultation system for improved antimicrobial selection. It returned an explanation as a subset of rules that led to the final result. Later, decision trees and bayesian networks were proposed and enabled to get interpretable models.

Recently, several approaches have relied on decision rules to provide explanations [47, 45, 56]. They are close to the way humans reason and can thus be used as explanations. Another type of models that has been extensively used by the XAI community is linear regression and its derivatives [3, 61, 30].

All these models are considered transparent assuming they are compact enough. Indeed, if they are too complex, even though their design favors explainability, they cannot provide an understandable explanation [1, 68].

2.2 Post-hoc Interpretability Methods for Images

In the last decade, deep neural networks have become very popular because of the great performance they achieve on image- and language-related tasks. While this performance is often unequalled by other types of models, their opacity and vulnerability to attacks [44] raised several issues. Multiple methods have been proposed to get clues about what a deep neural network, or more generically a black box, is doing.

Feature attribution methods, which aim at quantifying the impact of all features on the final decision provided by the model under study, have got a lot of attention: we distinguish model agnostic methods [69, 53, 16] from more specific ones relying on the generation of saliency maps [72, 85, 73, 59, 42, 74]. Another type of strategies consists in visualizing the patterns that activate a unit (a single neuron, channel or a complete layer) the most [21, 81, 62]. At the scope of the whole model, knowledge distillation [17, 11, 32, 24] enables to train a student model, which is usually explainable (cf. Section 2.1), with a teacher model, which is hardly interpretable. Some other approaches are based on concept activation vectors [41, 29, 27], which enable to assess how a user-defined concept contributes to a result, or on prototypes and criticisms [40], which are instances that are respectively highly representative or not representative at all of the dataset.

3 Theoretical Background

3.1 Fuzzy Logic

Fuzzy Logic and the fuzzy set theory have been introduced by Zadeh in [82]. It can be seen as an extension of Boolean logic that enables to manage imprecision. While a value is either true or false in Boolean logic, it can range from 0 (false) to 1 (true) in Fuzzy Logic.

In a universe $\mathcal{U}$, a fuzzy set F is characterized by a mapping $\mu_F : \mathcal{U} \to [0, 1]$. This mapping specifies in what extent each $u \in \mathcal{U}$ belongs to F and it is called *the membership function of* F.

If F is a non-fuzzy set (also known as *crisp* set), $\mu_F(u)$ is either 0, i.e. u is not a member of F, or 1, i.e. u is a member of F. In the following, we will use the expressions *non-fuzzy* and *crisp* interchangeably.

In [83], Zadeh defined a *linguistic variable* as a triplet (V, Δ_V, F_V) such as:

- V is the name of the variable,
- Δ_V is the domain on which V is defined,
- $F_V = \{F_1, F_2, ...\}$ is a finite collection of linguistic terms, which are formalized as fuzzy sets. Each of these linguistic terms qualifies V.

For example, let us consider the linguistic variable $(\text{“distance”}, [0; 100], \{F_{\text{short}}, F_{\text{long}}\})$ represented in Figure 2. Δ_V is here the domain of distances in metres. The fuzzy set F_{short} enables to characterize how short a distance is while F_{long} evaluates how long it is. For $\delta \in [0; 25]$, $\mu_{\text{short}}(\delta) = 1$ so a distance of δ metres is short. For $\delta \in [25; 75]$, the shortness is imprecise. The values of μ_{short} enable to quantify the vagueness of the definitions. For $\delta \in [75; 100]$, $\mu_{\text{short}}(\delta) = 0$ so the distance is not short. However, that does not necessarily mean that it is long. Indeed, there can be several intermediary linguistic terms in F_V between “short” and “long”, which would contribute to make the approach more expressive.

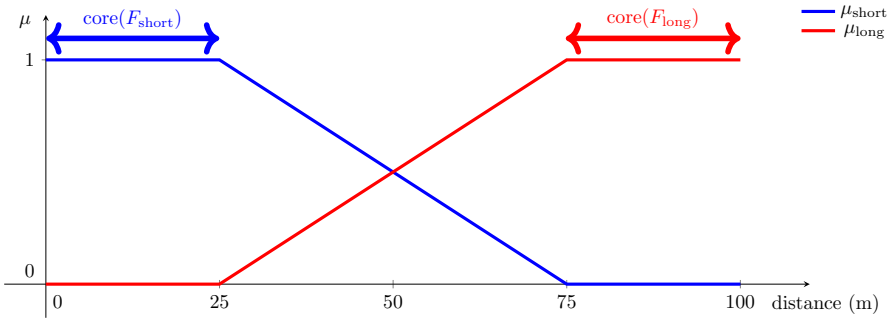


Figure 2: Representation of the linguistic variable $(\text{“distance”}, [0; 100], \{F_{\text{short}}, F_{\text{long}}\})$. The membership functions of the two fuzzy sets F_{short} and F_{long} are represented in blue and red respectively. The core [6] of F_{short} is $[0; 25]$ and the core of F_{long} is $[75; 100]$.

An elementary fuzzy proposition "V is A" is defined from a linguistic variable (V, Δ_V, F_V) with $A \in F_V$. For example, the fuzzy proposition "the distance is short" is assessed using the membership function μ_{short}. For a specific duration $\delta \in \Delta_V$, the truth value of this proposition is returned by $\mu_{\text{short}}(\delta)$ and it is interpreted as shown in the previous example.

3.2 Fuzzy Frequent Itemset Mining

Frequent itemset mining aims at extracting frequent patterns in a database. It has originally been introduced for performing *association rule learning* [2]. In such problems, the goal is to build rules that catch the frequent patterns in the database. The most common example of association rule learning is the *market basket problem*. In this problem, we have a dataset of transactions made by customers: each transaction contains items that one customer purchased. Based on this dataset, the objective is to extract rules that describe well the behaviour of consumers and relies on assessing which items are frequently bought together. In this example, an item belongs to a transaction or not, so a transaction can be represented as a crisp set of items. However, when the information in the database is vague, traditional frequent itemset mining methods are not well suited.

Fuzzy frequent itemset mining has been introduced to deal with such databases. It relies on the notion of *fuzzy formal context*, which is defined as a tuple $(\mathcal{T}, \mathcal{I}, \mathcal{R})$ such as

- $\mathcal{T}$ is a set of transactions,
- $\mathcal{I}$ is a set of items,
- $\mathcal{R} : \mathcal{T} \times \mathcal{I} \rightarrow [0; 1]$ is a dyadic fuzzy relation that expresses to which extent items belong to transactions.

Given a fuzzy formal context $(\mathcal{T}, \mathcal{I}, \mathcal{R})$, we can define the *support* of a set I of items [67]:

$$\forall I \subseteq \mathcal{I}, \text{support}(I) = \frac{\sum_{t \in \mathcal{T}} \min_{i \in I} \mathcal{R}(t, i)}{|\mathcal{T}|} \quad . \tag{1}$$

support(I) represents the frequency of I in the database. Thus, we can set a threshold S to assess if a set of items is *frequent* or not. We call this threshold the *minimum support* and all the sets I of items are considered frequent if, and only if, support$(I) \geq S$.

A frequent set I of items is *maximal* if, and only if, there exists no $I' \subseteq \mathcal{I}$ such that $I \subset I'$. Given a closure operator h defined on $\mathcal{I}$, I is *closed* if, and only if, $I = h(I)$. Maximal and closed frequent sets of items have interesting properties that our approach relies on (cf. Section 4.3).

3.3 Fuzzy Constraint Satisfaction Problems

A Constraint Satisfaction Problem (CSP) [58, 54, 78] consists in assigning some values to a set of variables that must respect a set of constraints. Many problems can be represented as a CSP like scheduling problems or the Golomb Ruler problem for placing sensors.

Dubois *et al.* [19] present an extension of CSPs to the fuzzy logic framework to deal with imprecise parameters and flexible constraints. This is called a *Fuzzy Constraint Satisfaction Problem* (FCSP), which is defined by:

- A set of variables $V = \{v_1, ..., v_m\}$.
- A set of domains $D = \{D_1, ..., D_m\}$ such as D_i is the range of values that can be assigned to v_i.
- A set of flexible constraints $C = \{c_1, ..., c_p\}$. Each constraint c_k is defined by a fuzzy relation $\mathcal{R}_k$ and by the set of variables V_k that are involved in it.

To solve a FCSP, a backtracking algorithm is applied. It starts with an empty set of instantiations and selects a variable $v_i \in V$ to instantiate. Then, it finds a value in the domain D_i that maintains the consistency of the current instantiation, regarding the set of constraints C. The steps are repeated until all the variables are instantiated. When a variable v_j has no more value to test, the algorithm backtracks and tries the next value of the previously instantiated variable.

To solve the problem faster, [19] proposed the FAC-3 algorithm, which is based on its crisp counterpart AC-3 [54]. The principle is to update the domains in D after each new instantiation of a variable (i.e.

after each iteration of the backtracking algorithm). This is performed by evaluating the constraints on the current instanciations and then discarding the values that are inconsistent.

FCSPs can be suitable to solve image annotation problems in which the labels and the objects to annotate are known (even if they are automatically detected, by a segmentation of the input image for instance) [77]. The intuition behind that approach is that such an annotation problem can be combinatorial and the labels are affected accordingly to each other, by opposition to individually like in classical approaches. In such a setting, the set of variables V corresponds to the objects we would like to instantiate. These variables share the same domain D that represents the regions obtained by segmentation. The constraints in C are defined by fuzzy relations between variables, which may involve groups of n objects ($n \geq 1$) [77].

4 Approach Description

As we saw in Section 2, explainable models should be favoured if the objective is to render the reasoning of the model. Such models usually rely on features that are interpretable, which makes the generation of a proper explanation easier. When we deal with raw input features, such as pixels in images, black box models are able to extract higher-level features that are more insightful for building an explanation. However, it is complicated to semantically characterize those automatically extracted features and to render the reasoning of these models.

In this section, we present our approach to build an explainable model that extracts higher-level features in images. These features are actually relations between entities in the input, like *A is to the left of B*. The model being explainable, we can directly express its reasoning and, since it handles interpretable relations between entities in the inputs, it is possible to generate an explanation that renders the reasoning of the model in a way understandable to humans.

4.1 Overview

The goal is to build a classification or annotation model by learning relations of interest from a given vocabulary on a training set. The explanations will be based on these learnt relations.

In the case of classification, we propose to rely on *decision rules*. They are closer to human reasoning [12] and to the language of reasoning [65], which is logic. Decision trees possess similar advantages, but they are known to be unstable [75, 20] because they can produce very different models for small changes in a dataset. Rules are usually inferred on the feature space but, as we wrote earlier, input features may not be suited to build explanations on. This is why we propose to extract relations that are associated to linguistic descriptions. A convenient framework for representing such relations is *Fuzzy Logic*, which enables to take into account both qualitative and quantitative information [82, 84]. Also, there exists an extensive literature about spatial fuzzy relations [10], which makes the whole framework well-suited for managing images.

For annotation, we propose to learn in a similar way constraints that will be then used to solve a *Fuzzy Constraint Satisfaction Problem* (FCSP) [77]. This constraint-based approach also satisfies the transparency requirement.

The knowledge about relations is brought by a vocabulary of potentially relevant relations that is set beforehand. Therefore, we can use regular datasets (where entities are labeled but not the relations between them) without labeling any relation in the instances. These relations are then assessed on entities that are part of the instances in the training set. Those entities are either directly provided in the dataset we use or we have to extract them using a segmentation algorithm.

Given a vocabulary of $n_{\mathcal{V}}$ relations $\mathcal{V}$ and a training set of n instances $\mathcal{D}$, the approach we propose consists in three main steps:

1. The relations from the vocabulary $\mathcal{V}$ are assessed on the training set $\mathcal{D}$. In Section 5, we present two strategies to prevent unnecessary computations to make the evaluation process faster.

2. The most relevant sets of relations are extracted according to the learning algorithm presented in Section 4.3.

3. Rules or constraints are generated from the relevant sets of relations so that classification or annotation can be performed. They can then be translated into a natural language explanation using the linguistic description associated to each relation in the rules/constraints, as shown in Section 4.4.

The whole approach is illustrated in Figure 3 in the case of image annotation.

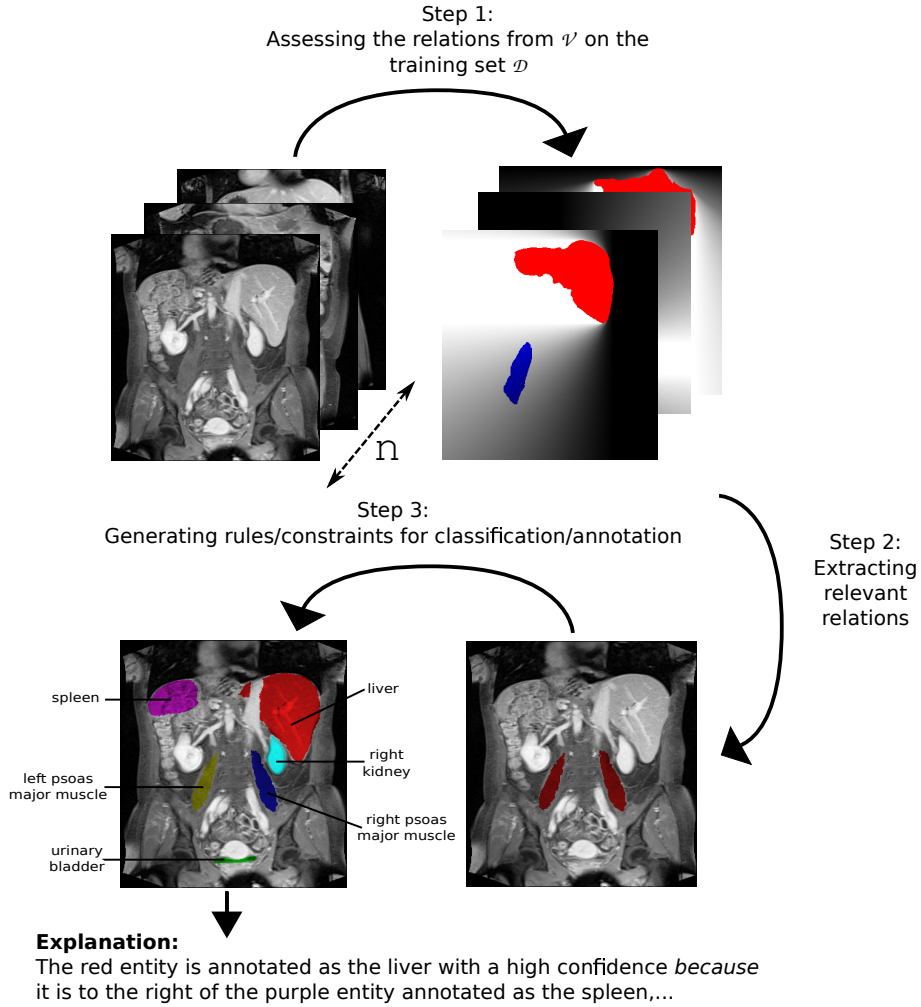


Figure 3: Illustration of our approach in the particular case of image annotation. $\mathcal{V}$ is a vocabulary of relations and $\mathcal{D}$ is the training set. For each annotation, an explanation is provided based on the constraints that directly led to this result.

4.2 Expressivity

The performance of a model strongly depends on the features it is fed with and on the way it handles feature values to compute a decision. If it is provided with few features, it may not be able to deal with a wide variety of situations. The same observation can be made if the reasoning of the model is too simple. In those two cases, the model is restricted by a lack of *expressivity*. In the context of XAI, the expressivity of the model also has an impact on the explanations it provides. Indeed, a lack of expressivity will lead to unconvincing explanations.

Our goal is to build a model able to perform classification or annotation and to provide an explanation to the decisions it makes. The relevance of the explanation depends on the relations that have been learnt and how the system uses them. That means that the original set of relations from which the most relevant ones are learnt has to be built wisely. A poor vocabulary could lead to bad decisions and irrelevant explanations. Thus, we need to ensure that our model is *expressive* enough to avoid this kind of situations.

Let us introduce the following notations:

- Let $\mathcal{V} = \{\mathcal{R}_1, \ldots, \mathcal{R}_{n_\mathcal{V}}\}$ be the vocabulary given for building a model. It is a set of $n_\mathcal{V}$ relations.
- Let $\alpha : \mathcal{V} \to \mathbb{N}$ be a function such as $\alpha(\mathcal{R})$ denotes the arity of the relation $\mathcal{R}$ for each $\mathcal{R}$ in $\mathcal{V}$.
- Let $\mathcal{X}$ be the space where instances are defined.
- Let $\mathbf{x}$ be an instance defined on the space $\mathcal{X}$.

- Let $\mathcal{O}_{\mathbf{x}} = \{o_{\mathbf{x},1}, \ldots, o_{\mathbf{x},K} \mid o_{\mathbf{x},i} \in \mathcal{X}, \forall i \in [\![1; K]\!]\}$ be a set of K entities in $\mathbf{x}$ that are defined on $\mathcal{X}$.
- Let $E_{\mathbf{x}}(\mathcal{V}) = \{\mathcal{R}(o_{\mathbf{x},1}, ..., o_{\mathbf{x},\alpha(\mathcal{R})}) \mid \mathcal{R} \in \mathcal{V}, (o_{\mathbf{x},1}, ..., o_{\mathbf{x},\alpha(\mathcal{R})}) \in \mathcal{P}(\mathcal{O}_{\mathbf{x}})\}$ be the set of all the relations in $\mathcal{V}$ evaluated on the entities in $\mathcal{O}_{\mathbf{x}}$. $\mathcal{P}(\mathcal{O}_{\mathbf{x}})$ is the power set of $\mathcal{O}_{\mathbf{x}}$.

$\mathcal{P}\big(E_{\mathbf{x}}(\mathcal{V})\big)$ is the set of all the possible explanations. It represents all the situations the model can express and it characterizes the expressivity of the model. The number of relations to evaluate for one instance $\mathbf{x}$ is:

$$|E_{\mathbf{x}}(\mathcal{V})| = \sum_{j=1}^{n_{\mathcal{V}}} \frac{|\mathcal{O}_{\mathbf{x}}|!}{\Big(|\mathcal{O}_{\mathbf{x}}| - \alpha(\mathcal{R}_j)\Big)!} \quad . \tag{2}$$

The time complexity for computing all the evaluations thus depends on the factorial of the number of entities present in $\mathcal{O}_{\mathbf{x}}$. Levesque and Brachman stated first that there is a dependency between the expressive power of a knowledge representation language and its computational tractability [48]. We encounter here a similar issue and we will propose two methods for pruning the evaluation space in Section 5.

4.3 Learning Relevant Relations and Descriptors

Being able to express relations between entities in an instance is a necessary but not sufficient condition for actually using these relations in an XAI. As an explanation should depict the reasoning of the system, it should rely only on the relations that are relevant enough for solving the given problem.

Since we aim at extracting the most relevant sets of relations for describing each class, we must first define what *relevance* means. Given a classification or annotation task, the relevance of a set of relations characterizes its appropriateness, in other words how it fits the classes, and how closely connected it is to the task, which means that it should help solving the problem induced by the task to perform.

In this work, we would like to determine the relations that are the most relevant to classes of entities. Thus, they should be representative of a class. However, that does not mean that they necessarily help to perform classification or annotation. For instance, the property that a tangerine is orange is relevant for classifying tangerines from bananas whereas it is not for classifying tangerines from oranges. While this property is important in describing the class *tangerine*, it is not sufficient for classification. As such, it should be part of a higher-level structure that represents a class and that is unambiguous. In other words, this structure should be both *descriptive* and *discriminative* [31, 46].

In this article, we call *descriptor* a set of relations that describes a class. So, given the vocabulary and the expressivity of the approach, we would like to extract the most relevant class descriptors. Relevant features should occur consistently whereas irrelevant features should occur inconsistently [39], which means that entities from one given class should share the same relevant relations. Thus, these relations should be frequent among the observations of entities from this class. As a consequence, we propose to rely on *frequent itemset mining* to extract the most frequent descriptors in the training set. *We assume here that relevance and frequency are equivalent.*

The limits of this assumption is that learning on few instances could be highly impacted by the presence of one or several outliers. Therefore, we will have to be careful about the way the training set is built. On the other hand, if the training set contains few outliers, it is theoretically possible to learn descriptors from few data because our model does not rely on a high number of parameters.

Since we are looking for the frequent subsets of relations of each class, we decide to carry out the learning phase using a *one vs all* approach. The learning is thus performed class by class. For one given class, instances should share the relations that are relevant, which means that they are highly correlated to each other. In order to take advantage of this property, we decided to rely on a fuzzy frequent itemset mining called Fuzzy Close [67]. This algorithm relies on a closure operator to take advantage of the high correlation between instances. This enables to return all the frequent sets of relations faster than other algorithms from the literature [4, 43, 35, 36, 63, 50, 51].

Let us represent a database of instances from the same class as a fuzzy formal context [7] $(\mathcal{T}, \mathcal{I}, \mathcal{R})$ such as:

- $\mathcal{T}$ is a set of instances,
- $\mathcal{I}$ is a set of evaluated relations,
- $\mathcal{R} : \mathcal{T} \times \mathcal{I} \to [0; 1]$ is a function that expresses the evaluation of relations in $\mathcal{I}$ on the instances in $\mathcal{T}$.

The goal is to extract all the frequent subsets of relations or, in other words, all the subsets $I \subseteq \mathcal{I}$ such that $\text{support}(I) \geq S$. Given a closure operator, the Fuzzy Close algorithm is able to do that in two steps [67]:

1. Determine all the frequent closed sets of relations. This is done in an iterative way: we first search all the frequent closed singleton of relations, then the frequent closed sets of size 2, and so on until we obtain all the frequent closed sets of relations.
2. Derive all the frequent subsets of relations from the frequent closed sets of relations.

When dealing with a dataset whose instances are highly correlated with each other, the number of closed frequent subsets of relations is much lower than the number of frequent subsets of relations. So, after the first step, we can derive frequent subsets of relations from just a few frequent closed subsets. That is why, in a setting where the dataset contains correlated data, this algorithm is faster than alternatives from the literature.

At the end of this step, we have for each class a set of descriptors that can be used for building rules or constraints.

4.4 Building the Model and Generating Explanations

The descriptors that we extracted at the previous step enable to build the model. For classification, we build fuzzy relational rules for each class. In the case of annotation, we generate constraints in order to define a FCSP.

4.4.1 Classification

The rules that are used in our approach involve fuzzy relations [60] and are called *fuzzy relational rules* [79, 80]

In our approach, we have already extracted relevant subsets of relations, which are the descriptors. Now, we are going to build fuzzy relational rules based on these.

In the Fuzzy Close algorithm, we derive frequent sets of items from the frequent closed sets of items. Consequently, there are many redundancies between frequent sets of items. We can get rid of those redundancies resorting to maximal sets of items. We also know that the set of maximal frequent sets of items is the same as the set of maximal frequent closed sets of items [64]. So we just need to find the maximal frequent closed subsets of relations among the frequent closed ones, which are given at the end of the first step of the Fuzzy Close algorithm (cf. Section 4.3). This task is straightforward. Let $\mathcal{Y}$ be the set of all possible labels where each label $y \in \mathcal{Y}$ is associated to a class of entities. For each $y \in \mathcal{Y}$, we get a set MFC_y of maximal frequent closed subsets of relations by applying the first step of the Fuzzy Close algorithm. In order to have discriminative descriptors, we remove from the descriptors all the relations that are common to several classes to get the set of discriminative descriptors MFC_y^* such as:

$$MFC_y^* = \left\{ I^* \mid I^* = I \setminus \bigcup_{y' \in \mathcal{Y} \setminus \{y\}} \left(\bigcup_{J \in MFC_{y'}} J \right), I \in MFC_y \right\} \tag{3}$$

Thus, we ensure that we get a descriptor that is both descriptive and discriminative. We assume here that the vocabulary has been set properly so that there is at least one $I^* \in MFC_y^*$ such that $I^* \neq \varnothing$.

Then, for each $y \in \mathcal{Y}$, we can build a rule for each descriptor I^* in MFC_y^* such as

$$\text{IF} \bigwedge_{\mathcal{R} \in I^*} \mathcal{R} \text{ THEN label} = y \tag{4}$$

In this work, conjunctions are computed as the minimum t-norm [23], which is the *min* operator.

Since there may be several descriptors for a given class, rules are actually aggregated following a fuzzy inference process [55]:

$$\forall y \in \mathcal{Y}, \mathbf{x} \in \mathcal{X}, \mu_y(\mathbf{x}) = \bigvee_{I^* \in MFC_y^*} \left(\bigwedge_{\mathcal{R} \in I^*} \mathcal{R} \right) \tag{5}$$

with $\mu_y(\mathbf{x})$ the membership degree of $\mathbf{x}$ to the class represented by label y and $\bigvee$ an aggregation operator, such as the supremum or the mean.

However, this definition does not rely on the support of the descriptors, which brings valuable information about their reliability. Indeed, descriptors do not all have the same support and so the rules they

entail should not all have the same weight in the final decision. Thus, we take into account the support of each descriptor to weigh the rules as proposed in fuzzy inference systems [55]:

$$\forall y \in \mathcal{Y}, \mathbf{x} \in \mathcal{X}, \mu_y(\mathbf{x}) = \bigvee_{I^* \in MFC_y^*} \left(\text{support}(I^*) \times \bigwedge_{\mathcal{R} \in I^*} \mathcal{R}\right) \tag{6}$$

where we can interpret $\mu_y(\mathbf{x})$ as the *confidence* in assigning the label y to the instance $\mathbf{x}$. Then, the predicted label $\hat{y}$ is the label associated to the highest confidence: $\hat{y} = \underset{y \in \mathcal{Y}}{\text{argmax}}\, \mu_y(\mathbf{x})$.

Depending on the value we set for the minimum support in the fuzzy frequent itemset mining step, some extracted relations might not be so relevant. Ideally, a few relations would be extracted when the value of the minimum support is equal to 1. In practice, it is not likely to happen. That is why we have to set this value carefully. If it is too high, there will not be enough frequent relations to discriminate classes. This is a case of *underfitting*. If it is too low, most relations may be considered as frequent while some of them are actually irrelevant, this is *overfitting*. Our strategy is to consider the minimum support as an hyperparameter of the problem. Thus, there are as many hyperparameters as there are classes. These hyperparameters will be set in a validation phase.

4.4.2 Annotation

As defined in Section 3.3, a constraint c is a pair $(\mathcal{R}, V)$ composed of a sequence V of variables, representing entities, and one relation $\mathcal{R}$ linking those variables. Thus, relations in descriptors can be directly translated into constraints.

The generation of a set of fuzzy constraints C for defining a FCSP is analogous to the generation of rules: it is performed class by class and it is based on extracting the set of maximal frequent closed subsets of relations using the Fuzzy Close algorithm. However, unlike rules, we do not prune the descriptors from the relations that are common to several classes. The idea is that we do not want to have too few constraints in our FCSP and that we should just ensure that there is no descriptor that describes several classes. The risk if too few constraints are learnt is that there may be too many highly consistent solutions. On the other hand, we could get no consistent solution if too many constraints are learnt, but this case should not happen if the various minimum supports (for each class) are set properly.

The difference with the previous case is that we only retain the descriptor with the largest cardinality. If there are several such descriptors, we select the one that has the greatest support in the training set. Let $y \in \mathcal{Y}$ be a label. Let I_y^M be the descriptor in MFC_y with the largest cardinality such that

$$I_y^M = \underset{I \in \mathcal{J}}{\text{argmax}} \left[\text{support}(I)\right] \text{ such as } \mathcal{J} = \{J \in MFC_y \mid |J| = \max_{P \in MFC_y} |P|\} \ . \tag{7}$$

As explained above, we assume the itemset of largest cardinality will be the most helpful for solving the problem since it enables to generate more constraints. Since constraints are fuzzy, we prefer having more constraints that may lead to a smaller degree of consistency than fewer constraints that may not be enough to solve the problem. In addition, the union of frequent maximal itemsets is not an acceptable choice since it is not a frequent itemset (otherwise it would be the only one maximal frequent itemset).

We know that each evaluated relation $\mathcal{R}$ in I_y^M links one or several classes of entities. Let $\Omega_\mathcal{R}$ be a set that contains those classes. In the definition of the FCSP, each variable is associated to a different class of entities. Therefore, for each item in I_y^M, we generate a constraint $(\mathcal{R}, V_{\Omega_\mathcal{R}})$ with $V_{\Omega_\mathcal{R}}$ the set of variables corresponding to the set of classes $\Omega_\mathcal{R}$.

After generating constraints for each relation and for each class, we have a set of constraints that can be used for defining and solving a FCSP. Given a new instance, the most consistent solution to this problem will lead to the annotation of every entity in the instance under study. As for rules, a *confidence* degree can be computed for a given annotation $y \in \mathcal{Y}$ as the product of the support of the descriptor I_y^M and the evaluation of the least consistent constraint in I_y^M.

There might be some constraints that appear several times. That means that several different classes produced the same constraint. That happens with symmetrical p-ary relations with $p > 1$. In that case, the set of constraints is reduced so that it contains this constraint only once.

4.4.3 Explanation Generation

We presented in the previous sections our methods for performing classification and annotation. Since our goal is to provide both a result and its explanation, we focus now on the generation of explanations

from the models we built. The explanation we would like to provide to the end-user is a sentence in natural language that explains how a result has been produced.

In particular, this step relies on using a *surface realizer*. In linguistics, a *realization* consists in generating a *surface form*, which is a correct sentence in a given natural language, from a more abstract representation in which the different components such as the subject or the verb are specified. Therefore, a surface realizer is a system that is able to take an abstract semantic representation as an input to generate a syntactically correct sentence. In this work, we decided to rely on SimpleNLG [25] for performing this task. This realization engine provides an API that is user-friendly and complete enough for the kind of explanations we would like to generate.

All the relations in the vocabulary are associated to a linguistic variable. That enables to express these relations in natural language. In addition, the evaluated relations that have been learnt to form class descriptors involve entities whose class is known. Thus, for each relation, we can generate a description in natural language. For example, let us consider a descriptor of a class of entities β containing the relation $\mathcal{R}_{\text{equal}}(\beta, \eta)$, η being another class of entities. It can be expressed in natural language such as "β equals η". Since each relation has a known and constant arity, automating the generation of an expression in natural language can be performed by associating a template to each relation. This template will be used in SimpleNLG, which will realize the corresponding sentence. In addition, since it is straightforward to convert a constraint into a relation (cf. the definition of a constraint in Section 3.3), constraints can be used to generate a natural language expression as we do with relations.

At a higher level, the surface realizer can express conjunctions of relations (classification) or constraints (annotation) and the confidence degree of the prediction being explained. The latter is represented as a linguistic variable representing the different levels of confidence: very high, high, average, low and very low. Then, a subordinating conjunction enables to link the prediction to its explanation and we get a complete explanation in natural language. For example, classifying an instance as a tangerine could lead to the following explanation:

This instance belongs to class "tangerine" with a high confidence because *entity A is orange, it is round and it is small.*

For classification, the result is returned by an aggregation of rules. When the aggregation operator is the supremum, only one rule contributes to the final result and thus generating the explanation is straightforward following the process described in the previous paragraph. When the aggregation operator is the mean, the basis for the explanation is the union of the antecedents of all the rules that are being aggregated.

In the case of annotation, one explanation is generated for each annotation on the basis of all the constraints the variable corresponding to the annotation was involved in.

Further works have been proposed for generating in natural language more complex explanations that can also express more logical combinations like disjunctions or negations, as in [5].

5 Optimizing the Evaluation of Relations

The approach we propose requires to evaluate fuzzy relations from a given vocabulary before learning the most frequent descriptors. At the end of the evaluation step, the system has a dataset that can be represented as a formal fuzzy context on which it can perform fuzzy frequent itemset mining (cf. Section 4.3).

The time complexity of this step directly depends on the relations that are evaluated. Keeping the same notations as in Section 4.2, the total number of evaluations to compute on the whole training set $\mathcal{D}$ is:

$$|E_{\mathcal{D}}(\mathcal{V})| = \sum_{i=1}^{n} \sum_{j=1}^{n_{\mathcal{V}}} \frac{|\mathcal{O}_{\mathbf{x}_i}|!}{\Big(|\mathcal{O}_{\mathbf{x}_i}| - \alpha(\mathcal{R}_j)\Big)!} \tag{8}$$

In particular, this quantity directly depends on the number of relations and their arities. Moreover, some relations may be *compute-intensive*, which makes the whole step longer. In order to keep it fast enough, two different types of strategies can be deployed:

- Speeding up the computation of relations based on their definition. The speed-up may be obtained by algorithm enhancement, distributed computation or approximating the result. It is a *local* optimization process (relation by relation).

- Filtering relations based on available knowledge, which is a *global* optimization process. Indeed, some evaluations of relations could be deduced from former evaluations if they satisfy some properties (symmetry, dependencies,...).

Both kinds of strategy are compatible since the first kind aims at reducing the computation time of one evaluation of a specific relation while the second kind enables to prevent unnecessary computations.

We propose two *global* heuristics that enable to prune the evaluation space and thus make the process faster. The first heuristic we present consists in not evaluating the relations that, after a few training instances, are bound to be infrequent. Each time a training instance has been studied, we compare the current support of each relation to the lowest value it must have to make the relation frequent. The other heuristic is based on the knowledge we have about the relations in the vocabulary. Such knowledge, like dependencies and implications between relations, can be represented in a graph. Then, this graph can be turned into a directed acyclic graph so that a topological sort can be obtained to get an order of evaluation on the relations in the vocabulary. Thus, only the necessary evaluations are computed.

5.1 Online Pruning of Infrequent Relations

Instances in the training set are evaluated one by one. Thus, the support of each relation can be updated each time an instance has been fully analyzed. Furthermore, to ensure that the frequency assumption holds true, *the minimum support S is always set to a value greater than or equal to at least 0.50.* We assume here that a relation that is, on average, fully satisfied in less than half of the instances in the training set is not representative of the class under study. This assumption is not restrictive since we do not expect to classify and explain an instance on the basis of such a relation. Indeed, this could harm the performance of the model and the reliability of the explanations. Thus, that enables to detect relations whose current support (after $k < n$ evaluated training instances) prevents their final support (after n evaluated training instances) to be greater than or equal to 0.5. This also presents the advantage of being independent of the vocabulary and the task to perform.

For example, let us consider the database $\mathcal{D}_{\text{fuzzy}}$ given in Table 1. $\mathcal{D}_{\text{fuzzy}}$ forms a fuzzy formal context $(\mathcal{T}, \mathcal{I}, \mathcal{R})$. With our assumption, we have $S \geq 0.5$. For any relation $i \in \mathcal{I}$ and $k \in [\![1; n]\!]$, let us note $S_k(\{i\})$ the support of i after k instances. For example, we have here

$$S_4(\{\mathcal{R}_2\}) = \frac{1}{4} \sum_{i=1}^{4} \mathcal{R}(t_i, \mathcal{R}_2) = 0.325$$

according to Equation (1). So, in the best case, if $\mathcal{R}(t_5, \mathcal{R}_2) = 1$, we would get

$$S_5(\{\mathcal{R}_2\}) = S(\{\mathcal{R}_2\}) = \frac{1}{5} \left(\sum_{i=1}^{4} \mathcal{R}(t_i, \mathcal{R}_2) + 1 \right) = 0.46 \ .$$

This value is lower than 0.5, so we can discard the relation $\mathcal{R}_2$ after the first four transactions. However, for the singleton $\{\mathcal{R}_3\}$, we need the five transactions to assess whether it is frequent or not.

Using the same reasoning as in the previous example, we can obtain the condition on which a relation can be discarded for any value of the minimum support. Let $i \in \mathcal{I}$ be a relation. Let k be an integer in $[\![1; n-1]\!]$. Let $S_k(\{i\})$ be the support of $\{i\}$ after k examples. Let S be the value of the minimum

Transactions	Items				
	$\mathcal{R}_1$	$\mathcal{R}_2$	$\mathcal{R}_3$	$\mathcal{R}_4$	$\mathcal{R}_5$
t_1	0.8	0.1	0.9	0.8	0
t_2	0	0.3	0.2	0	0.9
t_3	1	0.7	0.7	1	0.6
t_4	0	0.2	0	0.2	1
t_5	0.9	0.6	0.8	1	0.9

Table 1: The fuzzy database $\mathcal{D}_{\text{fuzzy}}$. For a minimum support greater than or equal to 0.5, we can know that $\{\mathcal{R}_2\}$ cannot be frequent after having processed the first four instances. However, for $\{\mathcal{R}_3\}$, we need its evaluation in t_5 to assess whether it is frequent or not.

support. $\{i\}$ is bound to be infrequent if, and only if, it verifies

$$S_k(i) < \frac{n(S-1)}{k} + 1 \tag{9}$$

We call B the bound $\frac{n(S-1)}{k} + 1$.

Proof. The support of $\{i\}$, $S_n(\{i\})$, can be expressed as

$$S_n(\{i\}) = \frac{kS_k(\{i\}) + \sum_{j=k+1}^{n} \mathcal{R}(t_j, i)}{n}, \forall k \in [\![1; n-1]\!] \ . \tag{10}$$

After k instances, the greatest possible value for $S_n(\{i\})$ is thus reached when $\mathcal{R}(t_j, i) = 1, \forall j \in [\![k+1; n]\!]$. As a consequence, $\{i\}$ is bound to be infrequent if, and only if

$$\frac{kS_k(\{i\}) + n - (k+1) + 1}{n} < S \tag{11}$$

which leads to

$$S_k(\{i\}) < \frac{n(S-1)}{k} + 1 \ . \tag{12}$$

□

B depends on S, on the size of the dataset n and on the current evaluation iteration k. That makes this heuristic independent of the task and the vocabulary. With our assumption, we have $S \geq 0.5$ and thus, in the worst case when $S = 0.5$, we have $B = 1 - \frac{n}{2k}$. If S is required to be greater than 0.5, it is possible to get a higher value of B, which enables to discard even more relations. In Figure 4, we represented B for several values of S and for $n = 30$. Values of S that are lower than 0.50 are not considered since it would mean that either our assumption that a relevant relation is frequent is false or that the vocabulary we use is too poor or not suited to the dataset. We chose n so that its magnitude is representative of the size of the training sets we dealt with in our experiments.

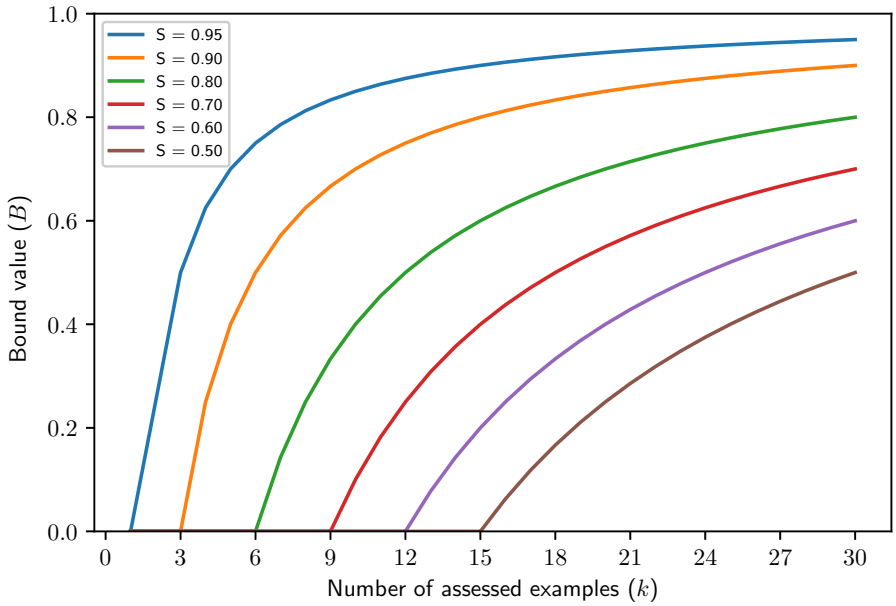


Figure 4: Curves representing the evolution of $B = \frac{n(S-1)}{k} + 1$ with the number of assessed examples k for a total number of examples $n = 30$. For the sake of clarity, only the positive values of B are displayed (if B is negative, no support can be discarded because it is always positive or null).

Overall, the greater the threshold S, the earlier we should be able to discard relations. When $S = 0.50$, we can start discarding relations once half of the training set has been treated. So, if a relation is never satisfied in the first half of the training set, we can avoid computing it for the second half.

While this heuristic takes advantage of the way frequent subsets of relations are learnt, another strategy is to exploit the properties of the relations in the vocabulary.

5.2 Knowledge-Based Ordering of the Evaluation of Relations

The previous heuristic does not require any knowledge about relations. When this knowledge is available, it is possible to conduct strategies that are fully compatible with the one presented in the previous section.

The heuristic we present in this section relies on the knowledge we can get from the definitions of the relations in the vocabulary. This knowledge enables to express *links* between relations. In this work, we are interested in three kinds of links: dependency, implication and symmetry. The principle is to propagate the information of evaluated relations (using the links between relations) to gain insight on non-evaluated relations. This materializes as an order of evaluation on relations with the relations conveying more information at the front.

5.2.1 Dependency

The first link between relations that we present is the *dependency*. Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two p-ary fuzzy relations defined on a space A. $\mathcal{R}_1$ is said to be *dependent on* $\mathcal{R}_2$ if, and only if, there exists a function $\text{dep} : [0;1] \to [0;1]$ such as

$$\forall e \in A^p, \mathcal{R}_2(e) = \text{dep}\big(\mathcal{R}_1(e)\big) \ .$$

where e is a set of p entities in A.

This link means that $\mathcal{R}_2(e)$ needs the result of $\mathcal{R}_1(e)$ to be computed. Thus, for each $e \in A^p$, $\mathcal{R}_1(e)$ should be computed before $\mathcal{R}_2(e)$. While this link between relations does not directly discard useless relations, it enables to prevent redundant computations. In addition, it is straightforward to set since it is directly given by the definitions of relations.

For example, let us consider a relation $\mathcal{R}_{\text{connected}} \colon A \times A \to [0;1]$ that assesses if two entities are connected. We could define a second relation, $\mathcal{R}_{\text{disconnected}} \colon A \times A \to [0;1]$, as the complement of the first relation to assess if two entities are disconnected. In that case, there is a dependency between those two relations.

5.2.2 Symmetry

The second type of link represents a specific property of relations: *symmetry*. Let $\mathcal{R}$ be a p-ary fuzzy relation defined on a space A such as $\mathcal{R} : A^p \to [0;1]$. $\forall (e_1, \ldots, e_p) \in A^p$, $\mathcal{R}$ is said to be *symmetric* if, and only if, any permutation of $(e_1, \ldots, e_p)$ does not modify the result of the evaluation of $\mathcal{R}$ on the set of entities $\{e_1, \ldots, e_p\}$.

For instance, with $\mathcal{R}_{\text{connected}}$, we have
$\mathcal{R}_{\text{connected}}(a_1, a_2) = \mathcal{R}_{\text{connected}}(a_2, a_1), \forall a_1, a_2 \in A$. So this relation is symmetric.

For any p-ary symmetric relation $\mathcal{R}$, for a set $\mathcal{O}_{\mathbf{x}}$ of p entities associated to an instance $\mathbf{x}$, the number of evaluations to compute involving $\mathcal{R}$ is (according to Section 4.2)

$$|E_{\mathbf{x}}(\mathcal{R})| = \frac{|\mathcal{O}_{\mathbf{x}}|!}{\Big(|\mathcal{O}_{\mathbf{x}}| - p\Big)!} \ . \tag{13}$$

Using the symmetry property, this can be divided by the number of permutations, $p!$, and we get

$$|E_{\mathbf{x}}(\mathcal{R})| = \frac{|\mathcal{O}_{\mathbf{x}}|!}{p!\Big(|\mathcal{O}_{\mathbf{x}}| - p\Big)!} \ . \tag{14}$$

Therefore, it will be important to determine which relations in the vocabulary are symmetric. In the special case of a dyadic relation ($p = 2$), we also say that this relation is *commutative*.

5.2.3 Implications

We consider here the implications between relations from the vocabulary. The idea is to propagate the result of one relation to another. For example, for two p-ary relations $\mathcal{R}_1$ and $\mathcal{R}_2$ defined on a space A, for any tuple of entities $e \in A^p$, if $\mathcal{R}_1 \Rightarrow \mathcal{R}_2$, then $\mathcal{R}_1(e)$ should be evaluated before $\mathcal{R}_2(e)$. The value of $\mathcal{R}_2(e)$ could then be deduced from $\mathcal{R}_1(e)$.

We are interested in the following four kinds of implications:

- The *logical implication* between two relations. If $\mathcal{R}_1 \Rightarrow \mathcal{R}_2$, then, for each $e \in A^p$, $\mathcal{R}_1(e)$ should be evaluated before $\mathcal{R}_2(e)$ because the evaluation of $\mathcal{R}_1(e)$ gives information about the evaluation of $\mathcal{R}_2(e)$. For instance, let us consider two dyadic relations: $\mathcal{R}_{\text{equal}}$ that characterizes whether two entities are equal or not and $\mathcal{R}_{\text{same size}}$ that characterizes if two entities have the same size. Since two equal entities have the same size, we have $\mathcal{R}_{\text{equal}} \Rightarrow \mathcal{R}_{\text{same size}}$. So it may be more convenient to evaluate $\mathcal{R}_{\text{equal}}$ first and to then deduce the value of $\mathcal{R}_{\text{same size}}$.
- The logical implication between a relation and the complement of another relation: $\mathcal{R}_1 \Rightarrow \overline{\mathcal{R}_2}$. For instance, if two entities are connected, then they cannot be disconnected.
- The logical implication between the complement of a relation and another relation: $\overline{\mathcal{R}_1} \Rightarrow \mathcal{R}_2$. For instance, if two entities are not connected, then they are disconnected.
- The logical implication between the complement of a relation and the complement of another relation: $\overline{\mathcal{R}_1} \Rightarrow \overline{\mathcal{R}_2}$. For instance, if two entities are not connected, then they cannot overlap with each other.

Since the relations we use are not Boolean, we have to use fuzzy implications. We use here the Łukasiewicz implication $\xrightarrow{L}$ [6], which is the same implication as in the fuzzy frequent itemset mining algorithm we apply [67]. To have an approach that is always consistent, we are only interested in situations where the fuzzy implication is equal to 1. The four following paragraphs are dedicated to establish properties for detecting relations verifying an implication. We also specify how to propagate the results for each implication.

Implication between two relations We consider here the implication $\mathcal{R}_1 \Rightarrow \mathcal{R}_2$. Let us recall that, $\forall a, b \in [0; 1]$, $\xrightarrow{L}(a, b) = \min(1 + b - a, 1)$. We want this implication to be equal to 1:

$$\min(1 + b - a, 1) = 1 \Leftrightarrow 1 + b - a \geq 1 \Leftrightarrow b \geq a \tag{15}$$

This result is also satisfied by other residual implications. Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two p-ary fuzzy relations defined on a space A. There exists a logical implication $\mathcal{R}_1 \Rightarrow \mathcal{R}_2$ if, and only if,

$$\forall e \in A^p, \mathcal{R}_2(e) \geq \mathcal{R}_1(e) \tag{16}$$

Proof. According to Equation (15), for each $e \in A^p$, we have
$\xrightarrow{L}\big(\mathcal{R}_1(e), \mathcal{R}_2(e)\big) = 1$ if, and only if, $\mathcal{R}_1(e) \leq \mathcal{R}_2(e)$. □

To be able to deduce the exact value of $\mathcal{R}_2(e)$ given $\mathcal{R}_1(e)$, the relation between $\mathcal{R}_1(e)$ and $\mathcal{R}_2(e)$ must be an equality. Since the upper bound on our fuzzy relations is 1, we have $\mathcal{R}_2(e) \leq 1$. If $\mathcal{R}_1(e) = 1$, then we have also $\mathcal{R}_2(e) \geq 1$ according to Section 5.2.3. That gives $\mathcal{R}_2(e) = 1$. So that implication enables to propagate the relations that are fully satisfied.

When $\mathcal{R}_1(e) < 1$, we have $\mathcal{R}_1(e) \leq \mathcal{R}_2(e)$, which does not enable to get the exact value of $\mathcal{R}_2(e)$. This may still be valuable information but we do not tackle that in this work.

Implication between a relation and the complement of another relation We consider here the implication $\mathcal{R}_1 \Rightarrow \overline{\mathcal{R}_2}$. We use the standard negation defined in [23] as the fuzzy complement. We can perform a reasoning similar to what we did for the previous implication.

Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two p-ary fuzzy relations defined on a space A. There exists a logical implication $\mathcal{R}_1 \Rightarrow \overline{\mathcal{R}_2}$ if, and only if,

$$\forall e \in A^p, \mathcal{R}_2(e) \leq 1 - \mathcal{R}_1(e) \tag{17}$$

Proof. For each $e \in A^p$, we have according to Equation (15)
$\xrightarrow{L}\big(\mathcal{R}_1(e), \overline{\mathcal{R}_2}(e)\big) = 1$ if, and only if, $\mathcal{R}_2(e) \leq 1 - \mathcal{R}_1(e)$. □

When $\mathcal{R}_1(e) = 1$, the exact value of $\mathcal{R}_2(e)$ is 0 since the fuzzy relations we use have a lower bound equal to 0. So we can propagate a fully satisfied relation to deduce that another relation is not satisfied at all.

Implication between the complement of a relation and another relation Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two p-ary fuzzy relations defined on a space A. There exists a logical implication $\overline{\mathcal{R}_1} \Rightarrow \mathcal{R}_2$ if

$$\forall e \in A^p, \mathcal{R}_2(e) \geq 1 - \mathcal{R}_1(e) \tag{18}$$

Proof. For each $e \in A^p$, we have according to Equation (15)
$\xrightarrow{L} \left(\overline{\mathcal{R}_1}(e), \mathcal{R}_2(e)\right) = 1$ if, and only if, $\mathcal{R}_2(e) \geq 1 - \mathcal{R}_1(e)$. □

When $\mathcal{R}_1(e) = 0$, the exact value of $\mathcal{R}_2(e)$ is 1 since the fuzzy relations we use have an upper bound equal to 1. So we can propagate a relation that is not satisfied at all to deduce that another relation is fully satisfied.

Implication between the complements of two relations Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two p-ary fuzzy relations defined on a space A. There exists a logical implication $\overline{\mathcal{R}_1} \Rightarrow \overline{\mathcal{R}_2}$ if

$$\forall e \in A^p, \mathcal{R}_2(e) \leq \mathcal{R}_1(e) \tag{19}$$

Proof. For each $e \in A^p$, we have according to Equation (15)
$\xrightarrow{L} \left(\overline{\mathcal{R}_1}(e), \overline{\mathcal{R}_2}(e)\right) = 1$ if, and only if, $\mathcal{R}_2(e) \leq \mathcal{R}_1(e)$. □

When $\mathcal{R}_1(e) = 0$, the exact value of $\mathcal{R}_2(e)$ is 0 since the fuzzy relations we use have a lower bound equal to 0. So we can propagate a relation that is not satisfied at all to deduce that another relation is not satisfied at all.

Table 2 recaps the four properties that we have just established and how to propagate the results for each type of implication.

Implication	Condition	Propagation
$\mathcal{R}_1 \Rightarrow \mathcal{R}_2$	$\mathcal{R}_2(e) \geq \mathcal{R}_1(e)$	If $\mathcal{R}_1(e) = 1$, then $\mathcal{R}_2(e) = 1$
$\mathcal{R}_1 \Rightarrow \overline{\mathcal{R}_2}$	$\mathcal{R}_2(e) \leq 1 - \mathcal{R}_1(e)$	If $\mathcal{R}_1(e) = 1$, then $\mathcal{R}_2(e) = 0$
$\overline{\mathcal{R}_1} \Rightarrow \mathcal{R}_2$	$\mathcal{R}_2(e) \geq 1 - \mathcal{R}_1(e)$	If $\mathcal{R}_1(e) = 0$, then $\mathcal{R}_2(e) = 1$
$\overline{\mathcal{R}_1} \Rightarrow \overline{\mathcal{R}_2}$	$\mathcal{R}_2(e) \leq \mathcal{R}_1(e)$	If $\mathcal{R}_1(e) = 0$, then $\mathcal{R}_2(e) = 0$

Table 2: This table shows the four types of implications between relations that our approach can process. Those implications enable to propagate the result of one relation to another. $\mathcal{R}_1$ and $\mathcal{R}_2$ are two p-ary fuzzy relations defined on a space A. e is a tuple of entities defined on A^p.

5.2.4 Graph Representation

The different links between relations that we use have been presented in the previous subsections. Each type of link can be represented in a graph using the edges defined in Table 3. This is a *labeled directed graph*.

A *labeled directed graph* is a triple (G, L, l) such that:

- $G = (V, E)$ is a *directed graph* such that:
  - V is a set of vertices,
  - E is a set of edges such that each edge in E is an ordered pair $(v_1, v_2) \in V^2$,
- L is a finite set of labels,
- $l : E \to \mathcal{P}(L)$ is a function that assigns a subset of labels to each edge in E.

For example, let us consider a vocabulary of relations $\mathcal{V} = \{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4, \mathcal{R}_5\}$, with, $\forall i \in [\![1; 5]\!]$, $\mathcal{R}_i : A^p \to [0; 1]$, and a set of links between relations $L = \{\text{d, i, ni, e, ne}\}$ such that:

- $l(\mathcal{R}_1, \mathcal{R}_2) = \{\text{ni, e}\}$,

- $l(\mathcal{R}_2, \mathcal{R}_1) = \{\text{d, ni, e}\}$,
- $l(\mathcal{R}_1, \mathcal{R}_3) = \{\text{ne}\}$,
- $l(\mathcal{R}_3, \mathcal{R}_1) = \{\text{d, i}\}$,
- $l(\mathcal{R}_3, \mathcal{R}_4) = \{\text{e}\}$,
- $l(\mathcal{R}_4, \mathcal{R}_3) = \{\text{e}\}$,
- $l(\mathcal{R}_5, \mathcal{R}_5) = \{\text{c}\}$.

The corresponding labeled directed graph is represented in Figure 5. The meaning of the edges in the graph are specified in Table 3.

The original goal of this strategy is to obtain an order on relations based on the way they are linked to each other. However, the labeled directed graph we generate here can be cyclic (like in Figure 5), which does not enable to define an order on relations.

Link	Notation	Corresponding edge
$\mathcal{R}_2$ depends on $\mathcal{R}_1$	d	$\mathcal{R}_2 \xrightarrow{\text{d}} \mathcal{R}_1$
$\mathcal{R}_1$ is symmetrical	c	$\mathcal{R}_1 \circlearrowleft \text{c}$
$\mathcal{R}_1 \Rightarrow \mathcal{R}_2$	i	$\mathcal{R}_1 \xrightarrow{\text{i}} \mathcal{R}_2$
$\mathcal{R}_1 \Rightarrow \overline{\mathcal{R}_2}$	e	$\mathcal{R}_1 \xrightarrow{\text{e}} \mathcal{R}_2$
$\overline{\mathcal{R}_1} \Rightarrow \mathcal{R}_2$	ni	$\mathcal{R}_1 \xrightarrow{\text{ni}} \mathcal{R}_2$
$\overline{\mathcal{R}_1} \Rightarrow \overline{\mathcal{R}_2}$	ne	$\mathcal{R}_1 \xrightarrow{\text{ne}} \mathcal{R}_2$

Table 3: Recap of the different kinds of link we consider between relations and their notation in the graph representation. The third column specifies how the corresponding edge is represented in a graph. $\mathcal{R}_1$ and $\mathcal{R}_2$ are two arbitrary p-ary fuzzy relations defined on a space A.

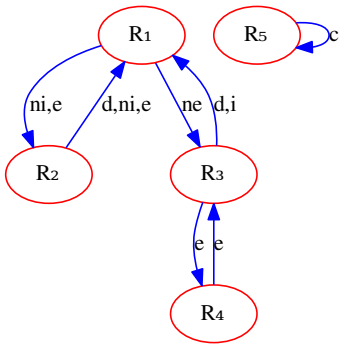


Figure 5: Example of labeled directed graph. The vertices are relations and the directed edges between vertices are labeled according to the links between relations. For each edge, its corresponding labels are to its right.

5.2.5 Extracting an Acyclic Graph

Our goal is to get an order on relations based on the knowledge graph representing them and their links. We propose to achieve that by getting rid of the cycles in the graph while keeping as much knowledge as possible to get a meaningful order. Then, once we have an acyclic graph, an order on its vertices can be obtained by topological sorting.

The knowledge graph representing the links between relations is directed. However, in order to obtain a topological sort of this graph, it must also be *acyclic*. We introduce Algorithm 1 and Algorithm 2 that enable to obtain a directed acyclic graph from the knowledge graph we have. The idea here is to build a new acyclic graph that preserves, insofar as possible, the information contained in the original one. Thus, we can get an order on relations that is faithful to the available knowledge.

Algorithm 1 is the main algorithm for converting our labeled directed graph LDG, which is the knowledge graph, into a directed acyclic graph DAG. The main loop starts at line 3 and loops over all the subgraphs g in LDG. At line 4, the conditional statement enables to check whether the subgraph has at least one edge or not. If the subgraph has no edge, then its vertices are added to V_{DAG} (line 5). Otherwise (from line 7), we loop over the edges of g (line 8). This loop first detects commutative relations and removes selfloops (line 9 to 12). Then, it extracts the dependency links and insert them in DAG (lines 13 to 17). We prioritize dependency over implications since the resulting relation cannot be computed without the relation it depends on. Then, the goal is to loop over LDG to extract implication links. The vertex used to start this exploration is selected from line 19 to line 25. We select one vertex in DAG, which is called *root*, that has the most outcoming edges among vertices without incoming edges (lines 20 and 21). *root* thus represents the relation involved in the most dependencies while not depending on any other relation. This is usually a generic relation so this choice enables to favour having this relation at the front of our order. If this is not possible, we select a random vertex in V_g (line 24). Then, we call the

Algorithm 1: algorithm that converts a labeled directed graph of relations into a directed acyclic graph of relations.

input : a labeled directed graph $LDG = (G_{LDG}, L, l)$ representing the links between relations
output: a directed acyclic graph DAG

$DAG \leftarrow (V_{DAG}, E_{DAG})$ such that $V_{DAG} = \varnothing$ and $E_{DAG} = \varnothing$
$LDG2 \leftarrow \text{copy}(LDG)$
forall subgraph $g = (V_g, E_g) \in G_{LDG}$ **do**
  if $E_g = \varnothing$ **then**
    $V_{DAG} \leftarrow V_{DAG} \cup V_g$
  end
  else
    forall edge $(v_1, v_2) \in E_g$ **do**
      if $v_1 = v_2$ **then**
        $V_{DAG} \leftarrow V_{DAG} \cup \{v_1^*\}$[a]
        $E_g \leftarrow E_g \setminus \{(v_1, v_1)\}$
      end
      else if d $\in l(v_1, v_2)$ **then**
        $V_{DAG} \leftarrow V_{DAG} \cup \{v_1, v_2\}$
        $E_{DAG} \leftarrow E_{DAG} \cup \{(v_2, v_1)\}$
        $E_g \leftarrow E_g \setminus \{(v_1, v_2)\}$
      end
    end
    if $\exists (v_1, v_2) \in V_g^2$ *such that* $(v_1, v_2) \in E_{DAG}$ **then**
      noParents $\leftarrow \{v \in V_{DAG} \cap V_g \mid \nexists (v_1, v_2) \in E_{DAG}$ such that $v_2 = v)\}$
      root $\leftarrow \underset{v \in \text{noParents}}{\text{argmax}} \left|\{(v, v') \in E_{DAG}\}\right|$
    end
    else
      root $\leftarrow$ randomly select v in V_g
    end
    `generateOtherLinks(`LDG`, `DAG`, `$LDG2$`, root)`
  end
end
return DAG

[a] The mark * is added to v_1 to specify that it represents a symmetrical relation.

Algorithm 2: algorithm that fills the directed acyclic graph with implication links. It represents the function `genrateOtherLinks`.

input:

- the labeled directed graph under study $LDG = (G_{LDG}, L, l)$,
- the directed acyclic graph $DAG = (V_{DAG}, E_{DAG})$ that is being built,
- a copy $LDG2$ of the original LDG (no edges have been removed),
- the vertex from which the algorithm starts, root.

```
1  visited ← ∅
2  Q ← queue(root)
3  while Q is not empty do
4      v_0 ← dequeue(Q)
5      neighbours ← {v ∈ V_LDG2 | (v_0, v) ∈ E_LDG2 or (v, v_0) ∈ E_LDG2}
6      if ∃(v_1, v_2) ∈ E_LDG such that v_1 = v_0 then
7          forall v_3 ∈ {v ∈ V_LDG | (v_0, v) ∈ E_LDG)} do
8              V_DAG ← V_DAG ∪ {v_3}
9              E_DAG ← E_DAG ∪ {(v_0, v_3)}
10             if DAG is cyclic then
11                 E_DAG ← (E_DAG \ {(v_0, v_3)}) ∪ {(v_3, v_0)}
12             end
13             E_LDG ← E_LDG \ {(v_0, v_3), (v_3, v_0)}
14         end
15     end
16     forall v ∈ neighbours do
17         if v ∉ visited then
18             visited ← visited ∪ {v}
19             Q ← queue(Q, v)
20         end
21     end
22 end
```

function `generateOtherLinks`(LDG, DAG, root) that is represented in Algorithm 2 and that completes DAG. Finally, once this is over, DAG is returned.

In Algorithm 2, we loop over the vertices of the graph using a breadth-first strategy starting from the vertex *root*. The objective is to complete the graph DAG with implication links (since dependencies and symmetries have already been filtered in Algorithm 1). At each iteration, the edges involving the node under consideration are added to the graph DAG. Since each edge (u, v) entails an edge (v, u) (by contraposition of the implication), we always select the direction that enables to ensure that the graph is acyclic (from line 9 to 12). Thus, a cycle cannot exist. Indeed, there cannot be a cycle of dependencies since it would mean that no relation can be computed first, which is absurd. Since selfloops are removed (line 11 in Algorithm 1) and we ensure that parallel edges of opposite directions cannot both be added to DAG (line 13 in Algorithm 2), this new directed graph is acyclic by construction, which was the primary goal of this new method. The following example illustrates it.

Let us consider the knowledge graph represented in Figure 5. It is a directed graph but it is not acyclic. We apply Algorithm 1 to get a directed acyclic graph so that we can perform topological sorting. The resulting graph is represented on Figure 6. The star in R_5^* means that it is a symmetric relation.

Once we get a directed acyclic graph DAG, we can perform topological sorting on all the subgraphs of DAG to get an order on the vertices, and so on the relations.

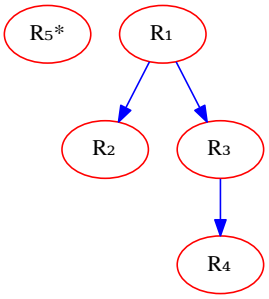


Figure 6: Example of directed acyclic graph that we get using Algorithm 1 on the knowledge graph displayed in Figure 5.

5.2.6 Global Method

In the previous subsections, we first studied the different kinds of links that may exist between the relations in the vocabulary. These links can be represented in a labeled directed graph where the vertices are the relations. Then, we saw how to convert this graph into a directed acyclic graph in order to perform topological sorting, which returns an order on vertices. Thus, the whole method consists in the following steps:

1. setting the links between relations from the vocabulary based on their definitions,
2. building a labeled directed graph LDG representing relations and how they are linked to each other,
3. converting LDG into a directed acyclic graph DAG using Algorithm 1,
4. obtaining a topological sort for each subgraph of DAG,
5. evaluating relations according to the topological sort we got in the previous step.

The following example depicts how we use the results of topological sorting to evaluate relations. For each tuple of entity, the order we got enables to propagate the values of evaluated relations to the ones that have not been evaluated yet. We extracted the links between relations in the vocabulary $\mathcal{V} = \{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4, \mathcal{R}_5\}$ and represented them in a labeled directed graph in Figure 5. Then, Algorithm 1 converted it into a directed acyclic graph that is displayed in Figure 6. In this acyclic graph, there are two subgraphs corresponding to the set of vertices $\{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4\}$ and $\{\mathcal{R}_5^*\}$. After performing topological sorting, we have the following results:

- $\mathcal{R}_1 \to \mathcal{R}_3 \to \mathcal{R}_4 \to \mathcal{R}_2$,
- $\mathcal{R}_5^*$.

where $\mathcal{R} \to \mathcal{R}'$ means that $\mathcal{R}$ should be evaluated before $\mathcal{R}'$. Since those two topological sorts are independent on each other, we can start by evaluating either $\mathcal{R}_1$ or $\mathcal{R}_5^*$, which is a symmetric relation. That means that, for a given tuple of entities in A^p, $\mathcal{R}_5^*$ should be evaluated only once. Let us now focus on the topological sort we got on the set of vertices $\{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4\}$. It means that $\mathcal{R}_1$ must be evaluated before $\mathcal{R}_3$, $\mathcal{R}_3$ must be evaluated before $\mathcal{R}_4$ and $\mathcal{R}_4$ must be evaluated before $\mathcal{R}_2$. Considering the original knowledge graph represented on Figure 5, this ordering of relations is justified by:

- $\mathcal{R}_2$ and $\mathcal{R}_3$ both depend on $\mathcal{R}_1$, so $\mathcal{R}_1$ should be computed before them.
- There are two parallel edges in opposite directions between $\mathcal{R}_3$ and $\mathcal{R}_4$ on Figure 5. Thus, there are two possibilities and, considering only these two relations, one could be computed before the other and vice versa. The way the graph DAG is completed depends on the node that is first visited. This is the node *root*, which is defined between lines 19 and 25 in Algorithm 1. If possible, this is the node that has the most relations dependent on itself. Consequently, in most cases, this node should not be a leaf. Here, the starting node was $\mathcal{R}_1$ and so $\mathcal{R}_3$ is visited before $\mathcal{R}_4$. That is why the edge between $\mathcal{R}_3$ and $\mathcal{R}_4$ is directed toward $\mathcal{R}_4$. We know that we have $\mathcal{R}_3 \Rightarrow \overline{\mathcal{R}_4}$, so if there exists $e \in A^p$ such that $\mathcal{R}_3(e) = 1$, then we get $\mathcal{R}_4(e) = 0$ without computing $\mathcal{R}_4(e)$.

In the next section dedicated to experiments, we assess both heuristics on classification and annotation tasks.

6 Experimental Results and Discussion

In this section, we present the two experiments we carried out to assess our approach. The first one deals with image classification while the second one tackles image annotation.

6.1 Image Classification on a Toy Dataset

6.1.1 Dataset

In this toy dataset of images, each example contains three *fuzzy* shapes: a square, a disk and an ellipse. These shapes correspond to the entities that will be handled in this experiment. Images from this dataset are divided into four classes. The difference between the classes is the spatial distribution of the fuzzy shapes. The shapes in each image of the same class are similarly spatially distributed. Examples from each class are shown in Figure 7. The dimensions and the fuzziness of each shape in each image vary independently of the class. In addition, four borderline examples have been added to each class in order to test the reliability of the system. Each class contains 44 images and so the whole dataset is composed of 176 examples. Each image contains 500×500 pixels.

Each shape is placed randomly in a restricted area of the image. Thus, for class 1, the square should be to the left of and slightly above the ellipse. The disk should be below and slightly to the left of the square. The ellipse should be to the top right of the disk. Classes 2, 3 and 4 are generated the same way and are rotated 90°, 180° and 270° counterclockwise respectively. In addition, for the sake of simplicity, ellipses always have the same orientation in every instance of each class.

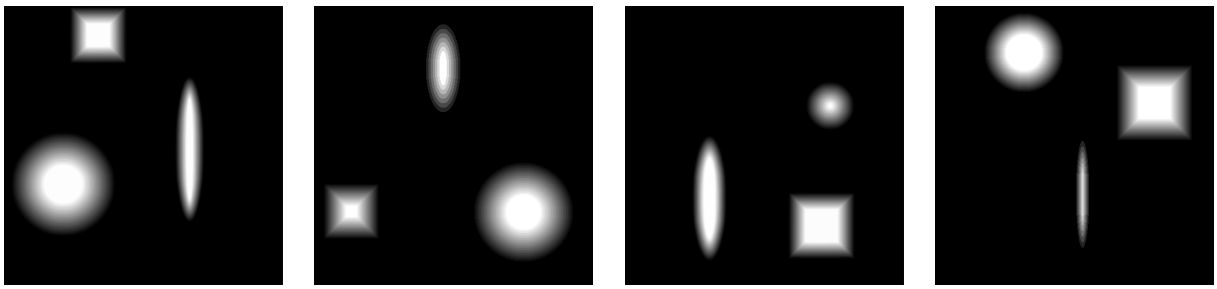
6.1.2 Vocabulary of Relations

Based on the way the dataset has been built, we limited the dyadic relations that are used in this experiment to directional relations. In particular, those are fuzzy directional dilations [9]. We use four of them: *to the left of*, *above*, *to the right of* and *below*. Figure 8 illustrates the evaluation process for evaluating the relation *ellipse to the right of disk*. Since the fuzzy dilation is a compute-intensive operator, generating fuzzy landscapes (cf. Figure 8c) is expensive. So we would like to prevent their generation when they are not necessary by using our two heuristics. When we do have to compute them, we relied on a fast implementation of this operator [66].

The unary relations we selected are shape-related. One of them assesses how close or far to a disk the shape of an entity is, another one assesses how close or far to a square the shape is and a third one assesses how close or far to an ellipse it is. We call them *is disk*, *is square* and *is ellipse*. They are based on an extension to fuzzy objects of the shape signature expressing the distance of boundary points to the centroid of the object [15]. It is based on averaging the signatures over the α-cuts [23] of the fuzzy object. We use this signature to build our three unary relations. Let χ be the signature of an entity. The relation *is disk* is defined as:

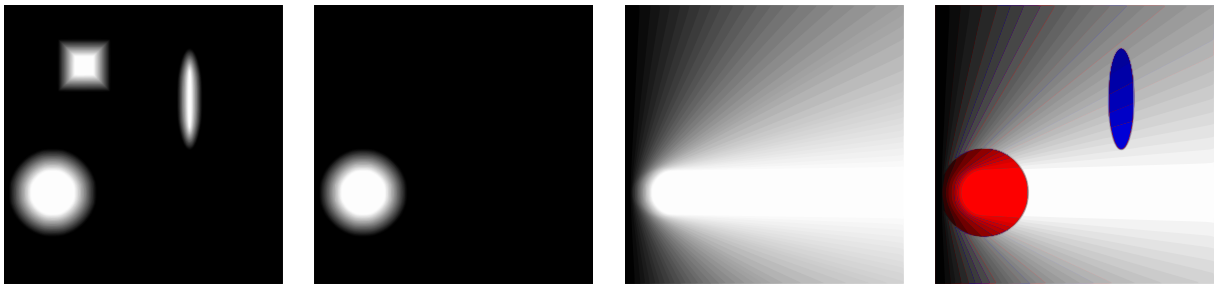
$$isDisk(\chi) = \begin{cases} 1 - \Delta & \text{if } \Delta \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

with $\Delta = \frac{\max(\chi) - \min(\chi)}{\text{mean}(\chi)}$. The relations *is square* and *is ellipse* are defined differently. They both return the absolute value of the correlation coefficient between χ and the signature χ_{ref} of a reference shape,



(a) Class 1 (b) Class 2 (c) Class 3 (d) Class 4

Figure 7: Examples from each class of the dataset used in the example of image classification in Section 6.1.



(a) Input (b) Disk (c) *To the right of* disk (d) Ellipse *to the right of* disk

Figure 8: Example of how a fuzzy directional relation is computed in an instance. Here, the goal is to compute the relation *ellipse to the right of disk*. Given an instance (Figure 8a), the disk, which is the reference object in the relation to evaluate, is extracted (Figure 8b). The fuzzy landscape *to the right of disk* can then be computed as the fuzzy dilation of the disk (Figure 8c). Finally, as illustrated in Figure 8d, the relation can be evaluated using a fuzzy pattern matching approach [14], which consists in computing the fuzzy degree of intersection between the fuzzy landscape and the ellipse.

such as:

$$isSquare(\chi) = |\text{corr}(\chi, \chi_{\text{ref}})| \tag{21}$$

with the reference shape a perfect square. *isEllipse* is defined the same way with a perfect ellipse as the reference shape.

Overall, we have 4 different spatial dyadic relations and 3 different unary relations. As there are 3 different entities in an instance, the total number of relations to evaluate per instance is equal to 33 according to Equation (2).

6.1.3 Ordering of Relations

In order to get an ordering of relations that enables to prevent some useless computations, the logical links between relations from the vocabulary are represented in a graph. This graph is displayed in Figure 9. The edges between directional relations hold because of the specific shapes of the entities present in the dataset.

Relying on the heuristic presented in Section 5.2, we applied Kahn's algorithm [38] to apply a topological sort and we get the following order:

$$\text{above} \rightarrow \text{below} \rightarrow \text{to the left of} \rightarrow \text{to the right of} \rightarrow \text{is disk} \rightarrow \text{is square} \rightarrow \text{is ellipse} \tag{22}$$

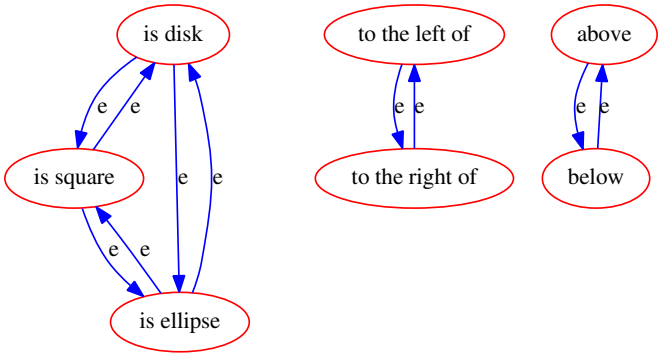


Figure 9: Graph representing the logical links between the relations in the vocabulary. There are three subgraphs with only one type of edge: e, which represents $\mathcal{R}_1 \Rightarrow \overline{\mathcal{R}_2}$.

6.1.4 Results

In this experiment, we evaluated the accuracy of the classifier using a 10-fold stratified cross validation strategy. Stratification enables to ensure that classes are uniformly distributed among all folds.

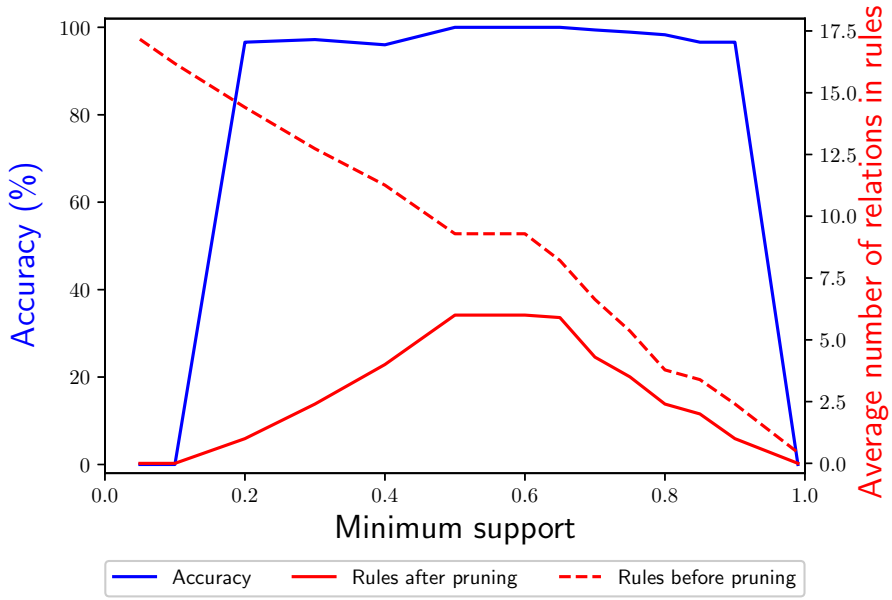


Figure 10: Plot displaying the performance of the model with respect to the value of the minimum support. We reach an accuracy of 100% between 0.5 and 0.65. The red dashed line represents the average number of relations in rules before they were pruned from relations common to other classes. The red plain line is its counterpart for rules that were pruned from these common relations.

Since the spatial arrangements of entities are similar for all classes and mainly differs from each other in the way they are rotated, we decided to set the minimum support at the same value for each class.

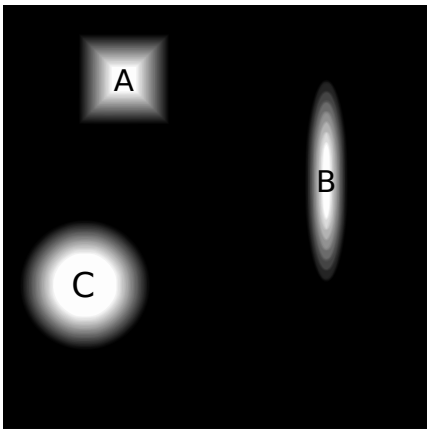
The results we got are displayed in Figure 10. For the sake of explaining the behaviour of the model, we studied the influence of the value of the minimum support over the whole range $[0; 1]$. We manage to reach an accuracy of 100% when the minimum support ranges from 0.5 to 0.65. This corresponds to the range of values where rules that were pruned from relations common to other classes reach a maximum length of 6. This observation seems logical: pruned rules contain relations that are class-specific, so longer pruned rules should have a better discriminative power.

We verify that the length of unpruned rules increases when the minimum support decreases. However, pruned rules do not follow the same trend. When the value of the minimum support is too low, many relations are considered frequent and thus rules from all classes share many common relations. Therefore, the pruning remove most of these relations, which are not relevant. That is actually a case of overfitting, since too many irrelevant relations are learnt.

On the other hand, when the value of the minimum support is too high, only few frequent relations are extracted. The risk is that those relations are common to all classes and thus cannot be used for classification. This is a case of underfitting. When all the relations in a rule are shared by some rules from other classes, the pruning turns this rule into an empty rule. This is why, in Figure 10, the average length of pruned rules is equal to 0 when the minimum support is set to 1.

Figure 11 and Figure 17, Figure 18 and Figure 19 in A show an example of explained classification for class 1, 2, 3 and 4 respectively. We can see that the most obvious relations are used for explaining the output. Moreover, one can notice in the example of Figure 11 that the relations *object B is below object A* and *object A is above object B* express the same situation. That seems obvious that if one of these two relations is used, the other one will be too. However, that may not always be the case due to the way these relations are computed. Indeed, when we apply a fuzzy dilation, the result depends on the shape of the reference object. So two reciprocal relations may not have the same evaluation. That is why there are slight differences that can affect the relations involved in the antecedent of the rule.

Our first heuristic, which consists in discarding infrequent relations to evaluate online, enabled us to avoid computing about 28% of the evaluations. For one given class, there are 44 instances. 33 relations are evaluated on each instance. That makes a total of 1452 relations to evaluate and our heuristic prevents 403 of them for classes 1 and 2, 401 for class 3 and 404 for class 4. In particular, it prevents useless computation of a few fuzzy landscapes, such as *above square* in class 1.



This instance belongs to class 1 with a very high confidence because:

- object C *is disk*,
- object A *is square*,
- object B *is ellipse*,
- object A *is above* object C,
- object A *is to the left of* object B,
- object C *is below* object A,
- object C *is to the left of* object B,
- object B *is to the right of* object C,
- object B *is to the right of* object A.

Figure 11: Example of explanation for an instance from class 1.

However, our second heuristic, which consists in using the logical links between relations, is not suited for this experiment. Indeed, there are no relations depending on others nor symmetrical ones. In the graph displayed in Figure 9, we represented the implications between a few relations. As we explained in Section 5.2, implications are useful when a relation is fully satisfied or not at all. Here, there are only implications that requires relations to be fully satisfied. Overall, only one relation has been evaluated to 1: this is the relation *object C below object A* in Figure 19. This enabled us to deduce that the relation *object C above object A* was equal to 0 before assessing it.

6.2 Image Annotation on a Medical Dataset

6.2.1 Dataset

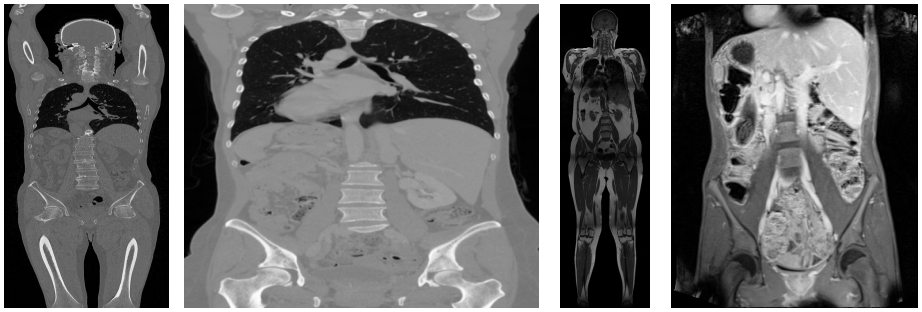
The original dataset we use here is called *Anatomy3* and was presented in [37]. It is composed of 391 CT and MRI images:

- CT images:
  - unenhanced CT scans of the whole body (CTwb);
  - enhanced CT scans of the whole trunk (CTce);
- MRI images:
  - unenhanced MRI scans of the whole body (MRIwb);
  - enhanced MRI scans of the abdomen (MRIce).

Figure 12 displays one example for each category of scan. Those are all 3D images that are actually the superposition of 2D slices. As we work on 2D images, we consider only slices in the following.

20 different organs are segmented among these images. Segmentation files are provided as binary images for each organ. Thus, the entities we deal with in this experiment are *not fuzzy*.

From these images, we created our own dataset for the purpose of this experiment. Since most images from the original dataset contain few segmented organs, we selected the instances containing the 9 following organs: the *liver*, the *spleen*, the *urinary bladder*, the *left* and *right kidneys*, the *left* and *right lungs* and the *left* and *right psoas major muscles*. That makes 35 images and 315 segments. These nine organs are represented in Figure 13. The goal will be to label them in each instance. This new dataset is small, which enables to assess how our model can perform by learning from few examples.



(a) CTwb (b) CTce (c) MRIwb (d) MRIce

Figure 12: Examples of the four types of scans in the dataset [37].

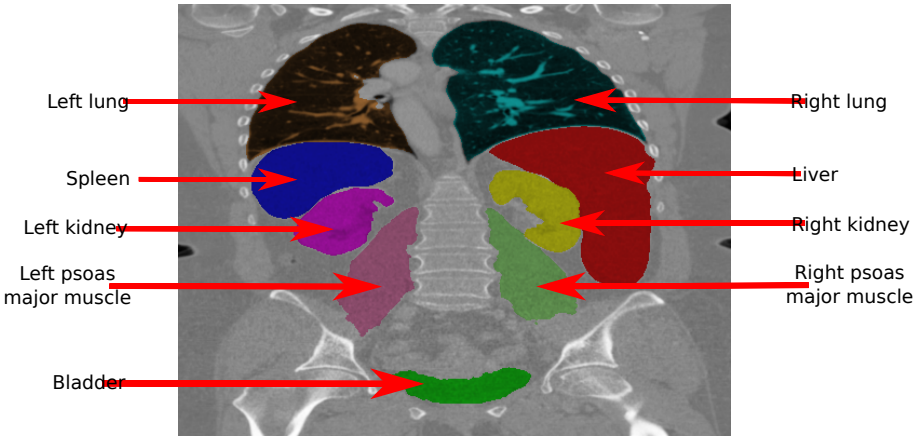


Figure 13: In this experiment, we consider the 9 colored organs in this figure.

6.2.2 Vocabulary of Relations

In this example, we used 9 relations. Four of them are the same directional relations [9] as we used in the previous experiment: *to the left of*, *above*, *to the right of* and *below*.

Four other relations are also directional. They express the same directions (left, right, above, below) but their fuzzy landscapes cover a smaller area of the image to express the following relations: *completely to the left of*, *completely to the right of*, *completely above* and *completely below*. They can be obtained by tweaking the structuring element in the definition of the relation [9]. Thus, these relations are less often satisfied and enable to express more accurate situations.

The last relation is the symmetry measure that is presented in [76]. The goal of this operator is to assess if two organs are symmetrical. It should be relevant since our dataset contains three pairs of organs: lungs, kidneys and psoas major muscles. This operator is applied on an image that includes only the two organs of interest for the relation to evaluate.

We have 9 fuzzy dyadic relations and each instance contains 9 entities. Applied on all entities, that makes 648 relations to assess per instance. These relations and how they are logically linked are represented in Figure 14. For our second heuristic, that leads to the following order of evaluation:

$$\begin{aligned}&\text{symmetrical to} \rightarrow \text{above} \rightarrow \text{completely above} \rightarrow \text{below} \rightarrow \text{completely below} \rightarrow \\ &\text{to the left of} \rightarrow \text{to the right of} \rightarrow \text{completely to the left of} \rightarrow \text{completely to the right of}\end{aligned} \tag{23}$$

6.2.3 Definition of the Fuzzy Constraint Satisfaction Problem

Since this is an annotation problem, our approach consists in solving a FCSP to annotate the organs. The set of variables and the set of domains of this FCSP are:

$$V = \{v_{\text{liver}}, v_{\text{spleen}}, v_{\text{bladder}}, v_{\text{r\_psoas}}, v_{\text{l\_psoas}}, v_{\text{r\_lung}}, v_{\text{l\_lung}}, v_{\text{r\_kidney}}, v_{\text{l\_kidney}}\} \tag{24}$$

$$D = \{D_{\text{liver}}, D_{\text{spleen}}, D_{\text{bladder}}, D_{\text{r\_psoas}}, D_{\text{l\_psoas}}, D_{\text{r\_lung}}, D_{\text{l\_lung}}, D_{\text{r\_kidney}}, D_{\text{l\_kidney}}\} \tag{25}$$

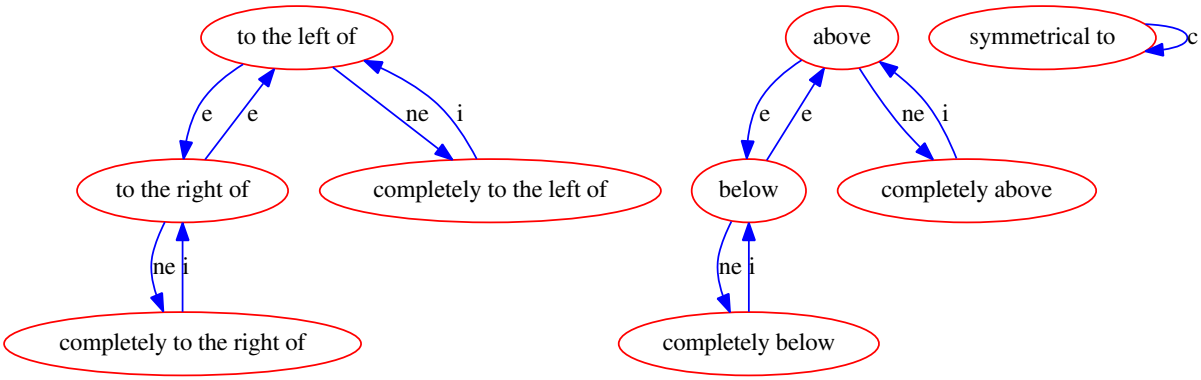


Figure 14: Graph representing the logical links between the relations handled by the model in this experiment.

where D_i is the set of all the possible values of v_i. The flexible constraints are generated from the frequent subsets of relations class by class as explained in Section 4. As a result, we get a set of constraints C. Furthermore, since every organ is unique, there cannot be identical labels in this problem. That means C has to be extended with constraints representing that two variables cannot be the same, which is the *AllDifferent* constraint:

$$\forall (v_i, v_j) \in V^2 \text{ such that } v_i \neq v_j, c_{i,j,\neq} = (v_i, v_j, \mathcal{R}_{\neq}) \tag{26}$$

where the constraint $c_{i,j,\neq}$ represents the relation $\mathcal{R}_{\neq}$ between two variables v_i and v_j. $\mathcal{R}_{\neq}(a, b)$ is a crisp relation that equals 0 if $a = b$ and 1 otherwise.

The FCSP is thus defined automatically. Then, for a given example, it can be solved as described in Section 3.3 using the FAC-3 algorithm, for filtering inconsistent domain values, and the backtracking algorithm, for exploring the possible solutions. At the end, the entities of interest have been labeled and an explanation can be produced based on the constraints that were derived from the descriptors. This explanation takes the form of a sentence in natural language so that the system achieves causability, which is essential in the medical domain [34].

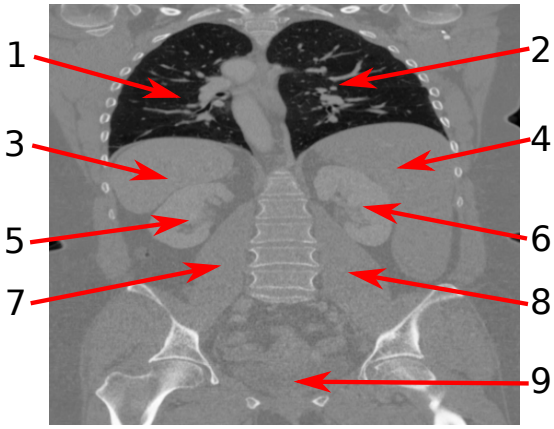
6.2.4 Results

The model we build with our approach relies on the frequent subsets of relations that are extracted. There are as many hyperparameters as labels and they correspond to the thresholds used for assessing the frequency of a subset of relations. Model selection is necessary to get optimized thresholds. However, we have a small dataset so performing hyperparameter tuning by splitting the dataset into a training set, a validation set and a test set is not convenient. In order to get a better performance estimation and an efficient tuning of hyperparameters, we resorted to *nested cross-validation* [13]: (1) an outer cross-validation is performed in which we get a training set and a test set for each iteration (this corresponds to a regular cross-validation), (2) an inner cross-validation is performed on the training set of the outer cross-validation to get an inner training set and a validation set for tuning hyperparameters. This enables to get an unbiased tuning of hyperparameters while also having the advantages of cross-validation.

In the inner cross-validation, hyperparameter tuning is performed using bayesian optimization over 20 iterations with a Gaussian process prior. The acquisition function is the expected improvement.

We first decided to perform a 5-fold cross-validation (it corresponds to the outer cross-validation in the description above). That means we have at each iteration a training set of 28 images while the test set is composed of 7 instances. The inner cross-validation, performed on the 28 examples in the training set, contains 3 folds. In this configuration, we reached 100% of accuracy. That means that, given accurate segments of the entities to label, our model performs well on this dataset.

One example of explained annotations is displayed in Figure 15. The length of the explanation directly depends on the number of constraints in which the organs were involved. Thus, all organs were involved in at least 4 constraints, except the bladder. Only one constraint involving this organ was learnt, which may not be ideal for generating a convincing explanation. We also notice that several reciprocal relations are present in explanations. This was expected based on our observations from the previous experiment (cf. Section 6.1.4). However, we get situations where two different direction accuracies are used: for example, in the explanation of entity 3 (the spleen) in Figure 15, we have:



Entity 1 is annotated as the left lung with a high confidence **because**:

- it is *completely to the left of* entity 2 (annotated as the right lung by the model),
- entity 2 (right lung) is *completely to its right*,
- it is *above* entity 3 (spleen),
- entity 3 (spleen) is *completely below* it,
- it is *above* entity 7 (left psoas),
- entity 7 (left psoas) is *completely below* it,
- entity 5 (left kidney) is *completely below* it.

Entity 2 is annotated as the right lung with a very high confidence **because**:

- it is *completely to the right of* entity 1 (left lung),
- entity 1 (left lung) is *completely to its left*,
- entity 3 (spleen) is *to its left*,
- entity 4 (liver) is *below* it,
- it is *above* entity 8 (right psoas),
- entity 8 (right psoas) is *completely below* it,
- entity 6 (right kidney) is *completely below* it,
- entity 9 (bladder) is *below* it.

Entity 3 is annotated as the spleen with an average confidence **because**:

- it is *completely to the left of* entity 4 (liver),
- entity 4 (liver) is *completely to its right*,
- it is *completely below* entity 1 (left lung),
- entity 1 (left lung) is *above* it,
- it is *to the left of* entity 6 (right kidney),
- entity 6 (right kidney) is *to its right*,
- entity 5 (left kidney) is *below* it,
- entity 7 (left psoas) is *below* it.

Entity 4 is annotated as the liver with a very high confidence **because**:

- it is *below* entity 2 (right lung),
- it is *completely to the right of* entity 3 (spleen),
- entity 3 (spleen) is *completely to its left*,
- entity 8 (right psoas) is *below* it,
- entity 6 (right lung) is *completely below* it.

Entity 5 is annotated as the left kidney with a high confidence **because**:

- it is *below* entity 3 (spleen),
- it is *to the left of* entity 6 (right kidney),
- it is *completely below* entity 1 (left lung),
- it is *above* entity 7 (left psoas.

Entity 6 is annotated as the right kidney with a high confidence **because**:

- it is *completely below* entity 4 (liver),
- it is *to the right of* entity 7 (left psoas),
- it is *completely below* entity 2 (right lung),
- it is *to the right of* entity 7 (spleen),
- entity 7 (spleen) is *to the left of* it,
- entity 5 (left kidney) is *to the left of* it,
- entity 8 (right psoas) is *below* it.

Entity 7 is annotated as the left psoas major muscle with an average confidence **because**:

- it is *is completely to the left of* entity 8 (right psoas),
- entity 8 (right psoas) is *completely to its right*,
- it is *completely below* entity 1 (left lung),
- entity 1 (left lung) is *above* it,
- it is *below* entity 3 (spleen),
- entity 6 (right kidney) is *to its right*,
- entity 5 (left kidney) is *above* it.

Entity 8 is annotated as the right psoas major muscle with a high confidence **because**:

- it is *completely to the right of* entity 7 (left psoas),
- entity 7 (left psoas) is *completely to its right*,
- it is *completely below* entity 2 (right lung),
- entity 2 (right lung) is *above* it,
- it is *below* entity 6 (right kidney),
- it is *below* entity 4 (liver).

Entity 9 is annotated as the bladder with a very high confidence **because**:

- it is *below* entity 2 (right lung).

Figure 15: Example of explained annotations.

Organ	Minimum support	Average number of constraints	Average confidence per annotation
Liver	0.99	5	0.99
Spleen	0.77	8.2	0.71
Bladder	0.96	1	0.97
Right lung	0.99	8	0.91
Left lung	0.99	7	0.89
Right kidney	0.99	7	0.88
Left kidney	0.99	4.2	0.88
Right psoas major muscle	0.91	6	0.86
Left psoas major muscle	0.85	6.8	0.70

Table 4: Table representing several results for each class of organ when performing a nested cross-validation with 5 folds in the outer loop. First, the minimum support associated to the learning of the frequent subsets of relations is given for each class. In the second column, the average number of constraints that we get for each class of organ at the end of the learning is displayed. Finally, we show the average confidence that we got for each annotation during the testing (accuracy of 100%).

- it is *completely below* entity 1 (left lung),
- entity 1 (left lung) is *above* it.

One may expect to get both *completely below* and *completely above* or both *below* and *above* in the explanation. This is due to the fact that, because of the shapes and sizes of the reference organs, the fuzzy landscape corresponding to *completely below the left lung* covers a bigger area of the image than the fuzzy landscape corresponding to *completely above the spleen*.

The constraints associated to the FCSP that enabled to annotate this instance are given in B.

The average values for the minimum supports associated to each organ are displayed in Table 4. We can notice that several organs have a minimum support of 0.99, which means that several relations are almost always fully satisfied. This happens because these organs satisfy very well the directional relations in the vocabulary. For example, the relation *left lung to the left of right lung* is equal to 1 in every instance of the dataset. Thus, for such organs, even a high minimum support enables to extract relations that are relevant to the problem under study.

Table 4 also displays the average number of constraints per organ over all the folds of the cross-validation. In particular, we notice that the spleen is the organ that is represented in the greater number of constraints. Even though each organ satisfies relations in a different way, this observation is consistent with the fact that the spleen has the lowest minimum support by a large margin. However, we can also notice that the right lung is involved on average in 8 constraints although its minimum support is equal to 0.99. This happens because it fully satisfies a few relations in every instance of the dataset, as described in the previous paragraph. Besides, there is always only one constraint involving the bladder, which is translated in the explanation of entity 9 in Figure 15. While we looked for the hyperparameters that enable to get the highest annotation performance, this may not always be compatible with the objective of explainability. This is one instance of the kind of tradeoff we encounter between performance and explainability.

The last column in Table 4 presents the average values of the confidences associated to the annotations of each organ. We notice that the two lowest average confidences correspond to the two classes of organ that have the lowest minimum supports. This is logical since the minimum support is one of the two factor in the computation of the confidence. All the organs that have a high minimum support and, on average, several constraints get a high confidence, which confirms the relevance of the constraints that were extracted during the learning phase. However, the case of the bladder is again interesting. It gets a high confidence although it is linked to only one constraint that may not seem relevant to a human (cf. Figure 15). We discuss this in further detail in Section 6.2.5.

We also investigated the number of training examples needed for our model to perform well. Using the nested cross-validation, we can evaluate the performance of the model for a number of training examples ranging from 17 to 34. Then, doing a reverse cross-validation (the training set and the test set are inverted), we can assess the performance of the model for a number of training examples ranging from

1 to 17. In that situation, instances are part of the test set in several iterations, but we ensure that the model has learnt from all the possible combinations of instances.

From 7 to 34 training examples, the model reaches an accuracy of 100%. Then, we get 99.6% for 5 training examples, and 99% with 3 and 2 training examples. These results show that the model can learn valuable information from a small dataset. In this experiment, the whole dataset does not contain any outlier (like a missing organ for example) so it is suited to learning from few data.

Before training, 21420 relations are evaluated over the whole dataset. We evaluated our first heuristic on these evaluations. Overall, this strategy enabled to prevent 32% of all the evaluations. We are especially interested in preventing expensive computations, which, given our vocabulary of relations, are:

- Fuzzy directional landscapes since they rely on the computation of a fuzzy dilation [66]. Since the evaluation of a relation is not expensive once the corresponding fuzzy landscape has been generated, we would like to avoid computing fuzzy landscapes if possible. For example, the fuzzy landscape corresponding to *to the right of the liver* is not generated if all the relations e *to the right of the liver* (e being any entity different from the liver) have been previously filtered by the heuristic. This is a strong constraint but in practice it is often satisfied when an entity is close to an edge of the image. For instance, if an entity e is always in the top left corner of the images, it is likely that the fuzzy landscapes *to the left of* e or *above* e will be dismissed at one point.

- Symmetries are also expensive since they rely on several operations that are compute-intensive [76].

Applying this heuristic, we managed to save 484 computations of the symmetry relation out of 1260 over the whole dataset (for a total of 2520 computations before taking into account the commutativity of this relation). For fuzzy landscapes, 220 computations were prevented out of 2520 over the whole dataset. Approximately 38% of the symmetries were prevented while about 9% of the fuzzy landscapes were. Preventing more symmetries than fuzzy landscapes is consistent with the fact that fuzzy landscapes are avoided if several relations are dismissed (against only one for the symmetry).

For the second heuristic, there are two different cases. First, the symmetry relation is commutative. That means that half of the evaluations of this relation can be saved. As mentioned in the previous paragraph, there were originally 2520 symmetries to evaluate and this number is thus reduced to 1260. For directional relations, the order presented in Equation (23) is followed to make the most of the logical links between these relations. It is important to note that once a relation is discarded with the first heuristic, we do not count it as a saved computation here. The results are presented in Table 5. This strategy enables to avoid computing the evaluation of 1636 relations, which represents 7.6% of the total number of evaluations. In particular, we notice that the logical link $\xrightarrow{\text{ne}}$ is more efficient than $\xrightarrow{\text{e}}$ on this dataset. This is due to the fact that there are more relations whose evaluation is equal to 0 than relations which are evaluated to 1: indeed, 3.8% of the evaluations are equal to 1 whereas about 50% of them are equal to 0. This high proportion of null evaluations also explains the efficiency of the first heuristic.

Overall, combining both heuristics, we manage to avoid computing about 40% of the total number of evaluations (8596 out of 21420 evaluations). In particular, it enables to avoid computing expensive relations like the symmetry or morphological directional relations. Coupled with the work presented in

Logical link	Number of prevented evaluations	Proportion among all evaluations
above $\xrightarrow{\text{ne}}$ completely above	367	1.7%
above $\xrightarrow{\text{e}}$ below	14	0.065%
below $\xrightarrow{\text{ne}}$ completely below	376	1.8%
to the left of $\xrightarrow{\text{ne}}$ completely to the left of	381	1.8%
to the left of $\xrightarrow{\text{e}}$ to the right of	113	0.53%
to the right of $\xrightarrow{\text{ne}}$ completely to the right of	385	1.8%
Total	1636	7.6%

Table 5: Table presenting the number of prevented evaluations using our second heuristic. Following the order of relations that we got, the results for each logical link are shown. Overall, this strategy enables to prevent 1636 evaluations, which represents 7.6% of the total number of evaluations.

[66], it has led to a significant improvement in the execution time of the evaluation step. However, we notice that the first heuristic, which aims at pruning infrequent relations online, is more efficient by a large margin.

6.2.5 Evaluation of Explanations

To evaluate explanations, we decided to perform a human-grounded evaluation [18] that relies on the method proposed by Baaj and Poli [5]. It consists in assessing a set of assertions that evaluate the language used in the explanations, the content and the form of the explanations, and how helpful to humans they are.

In this experiment, we asked 26 people, including one medical doctor, to answer the following assertions:

1. Explanations are simple and easy to read,
2. Explanations are convincing,
3. Data and explanations are sufficient to trust the system,
4. Explanations indirectly express the way the system reasons,
5. The length of the explanations is adequate,
6. It is difficult to read explanations until the end,
7. Explanations seem consistent,
8. Explanations are true.

Participants were asked to assess these assertions using the following Likert scale [49]: strongly disagree, disagree, undecided, agree, strongly agree. Participants also had the possibility to insert any comment for each assertion evaluation.

The results we got are displayed in Figure 16. Regarding the form of the explanations, 88% of the participants think that explanations are simple and easy to read. However, about 30% of them believe that it is difficult to read explanations until the end and 15% are undecided. That suggests that explanations may be too long, which is supported by the assessment of the length of the explanations. Indeed, half of the participants are not convinced by the length of the explanations. Thus, this is an area of improvement. In our approach, a higher minimum support should lead to shorter explanations but it also has a direct impact on the performance of the model. There is no clear way to favour one criterion over another.

Besides, several participants highlighted the redundancy in the explanations as a reason why they are difficult to read until the end. Although it is not the goal of this work, using synonyms or different sentence structures to break the monotony of the explanations could help.

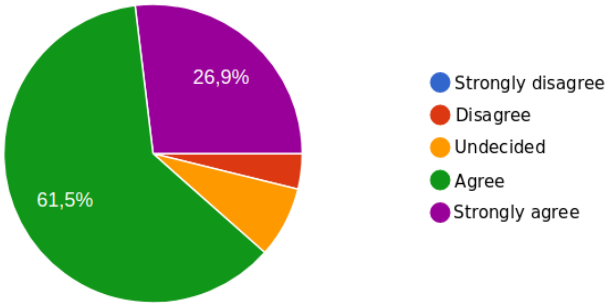
Most participants think that the explanations are convincing (69%), are true (65%), seem consistent (96%), render the reasoning of the system (81%) and enables to trust it (62%). Those are all good points since a few of the goals of explainable approaches are to increase trust in AIs and to make their reasoning more transparent.

However, while 96% of the participants think that the explanations seem consistent, fewer of them (although still a majority) find them convincing (69%) and trustworthy (62%). That means that consistency is not enough to ensure that people will trust the model. For example, it was pointed out that the explanation for the annotation of the bladder is consistent but is not convincing. In particular, the doctor that answered the survey said that explanations should favour local relations. This is not the case in the explanation of the annotation of the bladder since it relies on a relation between the bladder and the right lung. This may be a limit of our assumption that relevance is equivalent to frequency. It also means that our vocabulary lacks the relevant local relations that could explain this annotation better.

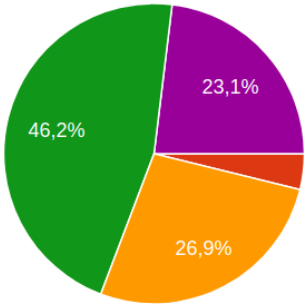
Furthermore, a few participants were confused because the order in which the entities were annotated is unknown, which harms the trustworthiness of the model. While the trace of the resolution of the FCSP could be made clear, we are not sure it would increase the trust in the system, especially for people who do not have any knowledge about CSPs.

Overall, this survey shows that most participants are convinced by the explanations and they understand the logic of the model. It also enabled to highlight the areas of improvement, such as the length of the explanations, their redundancy and the use of non-local relations.

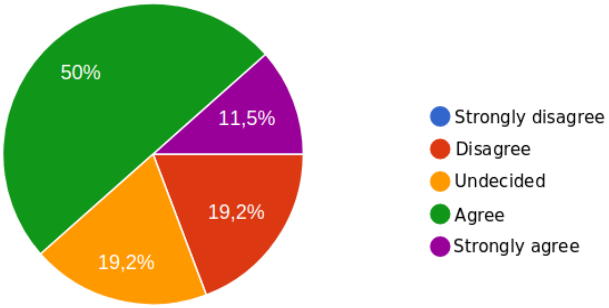
- Explanations are simple and easy to read:



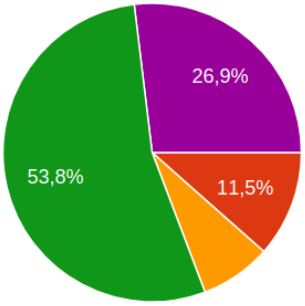
- Explanations are convincing:



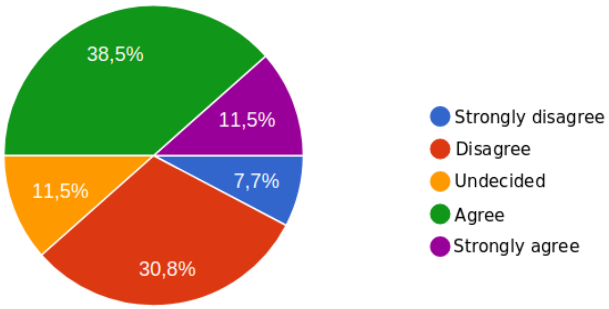
- Data and explanations are sufficient to trust the system:



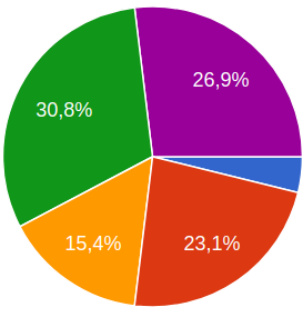
- Explanations indirectly express the way the system reasons:



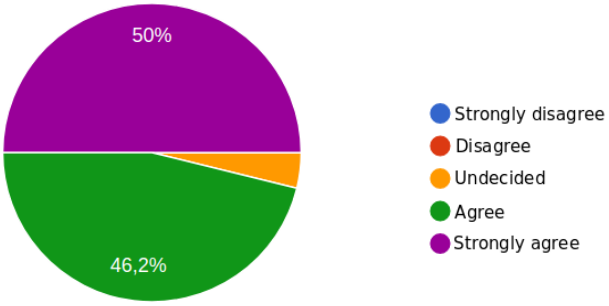
- The length of the explanations is adequate:



- It is difficult to read explanations until the end:



- Explanations seem consistent:



- Explanations are true:

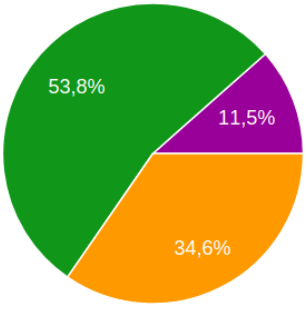


Figure 16: Answers to the eight assertions.

7 Conclusion

In this paper, we first introduced a fully explainable approach for performing image classification or annotation in the context of high-stake applications. It relies on learning the most relevant relations in a training set to build fuzzy rules or constraints. Since the reasoning is transparent and the relations are all associated to a linguistic variable, an explanation in natural language can be generated for each decision provided by the model. As evaluating all relations is too expensive, we proposed two heuristics that take advantage of the characteristics of the relations and of the learning algorithm to prevent unnecessary computations. Our experiments show that, given a segmentation of the input, the model our approach builds is able to successfully perform classification or annotation and to generate consistent and convincing explanations.

As future work, we aim at enriching our model with rules that can express disjunctions, negation or existence to deal with a wider range of situations. Also, when we extract the most frequent subsets of relations, we would like to insert a criteria that enables to specify the intended length of explanations. Finally, we think our approach is generic and can be extended to other kinds of inputs, such as time series.

References

[1] *An Evaluation of the Human-Interpretability of Explanation*, 2018.

[2] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, pages 207–216, 1993.

[3] D. Alvarez Melis and T. Jaakkola. Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems 31*, pages 7775–7784, 2018.

[4] W-H. Au and K. C. C. Chan. An effective algorithm for discovering fuzzy rules in relational databases. In *Fuzzy Systems Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, volume 2, pages 1314–1319, 1998.

[5] I. Baaj and J-P. Poli. Natural language generation of explanations of fuzzy inference decisions. In *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6, 2019.

[6] M. Baczynski, B. Jayaram, S. Massanet, and J. Torrens. *Fuzzy Implications: Past, Present, and Future*, pages 183–202. Springer Berlin Heidelberg, 2015.

[7] R. Belohlavek. *Fuzzy relational systems: foundations and principles*, volume 20. Springer Science & Business Media, 2012.

[8] I. Biederman. *On the Semantics of a Glance at a Scene*. 1981.

[9] I. Bloch. Fuzzy relative position between objects in image processing: a morphological approach. *IEEE transactions on pattern analysis and machine intelligence*, 21(7):657–664, 1999.

[10] I. Bloch. Fuzzy spatial relationships for image processing and interpretation: a review. *Image and Vision Computing*, 23(2):89–110, 2005.

[11] C. Buciluǎ, R. Caruana, and A. Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541, 2006.

[12] R. M. J. Byrne and P. N. Johnson-Laird. 'if'and the problems of conditional reasoning. *Trends in Cognitive Sciences*, 13(7):282–287, 2009.

[13] G. C. Cawley and N. L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(Jul):2079–2107, 2010.

[14] M. Cayrol, H. Farreny, and H. Prade. Fuzzy pattern matching. *Kybernetes*, 11(2):103–116, 1982.

[15] J. Chanussot, I. Nyström, and N. Sladoje. Shape signatures of fuzzy star-shaped sets based on distance from the centroid. *Pattern Recognition Letters*, 26(6):735–746, 2005.

[16] J. Chen, L. Song, M. Wainwright, and M. Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 883–892, 2018.

[17] M. Craven and J. W. Shavlik. Extracting tree-structured representations of trained networks. In *Advances in neural information processing systems*, pages 24–30, 1996.

[18] F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[19] D. Dubois, H. Fargier, and H. Prade. Possibility theory in constraint satisfaction problems: Handling priority, preference and uncertainty. *Applied Intelligence*, 6(4):287–309, 1996.

[20] K. Dwyer and R. Holte. Decision tree instability and active learning. In *European Conference on Machine Learning*, pages 128–139, 2007.

[21] D. Erhan, Y. Bengio, A. Courville, and P. Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[22] E. A. Feigenbaum, B. G. Buchanan, and J. Lederberg. On generality and problem solving: A case study using the dendral program. 1970.

[23] J. C. Fodor and I. J. Rudas. *Basics of Fuzzy Sets*, pages 159–170. Springer Berlin Heidelberg, 2015.

[24] N. Frosst and G. Hinton. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*, 2017.

[25] A. Gatt and E. Reiter. Simplenlg: A realisation engine for practical applications. In *Proceedings of the 12th European Workshop on Natural Language Generation*, pages 90–93, 2009.

[26] P. Geurts. Pattern extraction for time series classification. In *Principles of Data Mining and Knowledge Discovery*, pages 115–127, 2001.

[27] A. Ghorbani, J. Wexler, J. Y. Zou, and B. Kim. Towards automatic concept-based explanations. In *Advances in Neural Information Processing Systems*, pages 9273–9282, 2019.

[28] A. González, R. Pérez, Y. Caises, and E. Leyva. An efficient inductive genetic learning algorithm for fuzzy relational rules. *International Journal of Computational Intelligence Systems*, 5(2):212–230, 2012.

[29] M. Graziani, V. Andrearczyk, and H. Müller. Regression concept vectors for bidirectional explanations in histopathology. In *Understanding and Interpreting Machine Learning in Medical Image Computing Applications*, pages 124–132. Springer, 2018.

[30] T. Hastie and R. Tibshirani. Generalized additive models. *Statistical Science*, pages 297–310, 1986.

[31] L. A. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell. Generating visual explanations. In *European Conference on Computer Vision*, pages 3–19, 2016.

[32] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[33] A. Holzinger. Interactive machine learning for health informatics: when do we need the human-in-the-loop? *Brain Informatics*, 3(2):119–131, 2016.

[34] A. Holzinger, G. Langs, H. Denk, K. Zatloukal, and H. Müller. Causability and explainability of artificial intelligence in medicine. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(4):e1312, 2019.

[35] T-P. Hong, C-S. Kuo, and S-C. Chi. Mining association rules from quantitative data. *Intelligent data analysis*, 3(5):363–376, 1999.

[36] T-P. Hong, C-S. Kuo, and S-L. Wang. A fuzzy aprioritid mining algorithm with reduced computational time. *Applied Soft Computing*, 5(1):1–10, 2004.

[37] O. Jimenez-del Toro, H. Müller, M. Krenn, K. Gruenberg, A. A. Taha, M. Winterstein, I. Eggel, A. Foncubierta-Rodríguez, O. Goksel, A. Jakab, et al. Cloud-based evaluation of anatomical structure segmentation and landmark detection algorithms: Visceral anatomy benchmarks. *IEEE transactions on medical imaging*, 35(11):2459–2475, 2016.

[38] A. B. Kahn. Topological sorting of large networks. *Commun. ACM*, 5(11):558–562, November 1962.

[39] R. T. Kellogg. Feature frequency and hypothesis testing in the acquisition of rule-governed concepts. *Memory & Cognition*, 8(3):297–303, 1980.

[40] B. Kim, R. Khanna, and O. O. Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in Neural Information Processing Systems 29*, pages 2280–2288, 2016.

[41] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International Conference on Machine Learning*, pages 2673–2682, 2018.

[42] P-J. Kindermans, K. T. Schütt, M. Alber, K-R. Müller, D. Erhan, B. Kim, and S. Dähne. Learning how to explain neural networks: Patternnet and patternattribution. *arXiv preprint arXiv:1705.05598*, 2017.

[43] C. M. Kuok, A. Fu, and M. H. Wong. Mining fuzzy association rules in databases. *ACM Sigmod Record*, 27(1):41–46, 1998.

[44] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[45] H. Lakkaraju, S. H. Bach, and J. Leskovec. Interpretable decision sets: A joint framework for description and prediction. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1675–1684, 2016.

[46] M-J. Lesot, M. Rifqi, and B. Bouchon-Meunier. Fuzzy prototypes: From a cognitive view to a machine learning principle. In *Fuzzy Sets and Their Extensions: Representation, Aggregation and Models*, pages 431–452. Springer, 2008.

[47] B. Letham, C. Rudin, T. H. McCormick, D. Madigan, et al. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3):1350–1371, 2015.

[48] H. J. Levesque and R. J. Brachman. Expressiveness and tractability in knowledge representation and reasoning 1. *Computational intelligence*, 3(1):78–93, 1987.

[49] R. Likert. A technique for the measurement of attitudes. *Archives of psychology*, 1932.

[50] C-W. Lin, T-P. Hong, and W-H. Lu. An efficient tree-based fuzzy data mining approach. *International Journal of Fuzzy Systems*, 12(2):150–157, 2010.

[51] C-W. Lin, T-P. Hong, and W-H. Lu. A two-phase fuzzy mining approach. In *Fuzzy Systems (FUZZ), 2010 IEEE International Conference on*, pages 1–5, 2010.

[52] R. K. Lindsay, B. G. Buchanan, E. A. Feigenbaum, and J. Lederberg. Dendral: A case study of the first expert system for scientific hypothesis formation. *Artificial Intelligence*, 61(2):209 – 261, 1993.

[53] S. M. Lundberg and S-I. Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30*, pages 4765–4774, 2017.

[54] A. K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8(1):99 – 118, 1977.

[55] L. Magdalena. *Fuzzy Rule-Based Systems*, pages 203–218. Springer Berlin Heidelberg, 2015.

[56] D. M. Malioutov, K. R. Varshney, A. Emad, and S. Dash. Learning interpretable classification rules with boolean compressed sensing. In *Transparent Data Mining for Big and Small Data*, pages 95–121, 2017.

[57] T. Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

[58] U. Montanari. Networks of constraints: Fundamental properties and applications to picture processing. *Information sciences*, 7:95–132, 1974.

[59] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K-R. Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.

[60] S. Montes, I. Montes, and T. Iglesias. *Fuzzy Relations: Past, Present, and Future*, pages 171–181. Springer Berlin Heidelberg, 2015.

[61] J. A. Nelder and R. W. M. Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135(3):370–384, 1972.

[62] C. Olah, A. Mordvintsev, and L. Schubert. Feature visualization. *Distill*, 2017. https://distill.pub/2017/feature-visualization.

[63] S. Papadimitriou and S. Mavroudi. The fuzzy frequent pattern tree. In *The WSEAS International Conference on Computers*, pages 1–7, 2005.

[64] N. Pasquier, Y. Bastide, R. Taouil, L. Lakhal, et al. Efficient mining of association rules using closed itemset lattices. *Information systems*, 24(1):25–46, 1999.

[65] D. Pedreschi, F. Giannotti, R. Guidotti, A. Monreale, S. Ruggieri, and F. Turini. Meaningful explanations of black box ai decision systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9780–9784, 2019.

[66] R. Pierrard, L. Cabaret, J-P. Poli, and C. Hudelot. Simd-based exact parallel fuzzy dilation operator for fast computing of fuzzy spatial relations. In *Proceedings of the 2020 Sixth Workshop on Programming Models for SIMD/Vector Processing*, pages 1–8, 2020.

[67] R. Pierrard, J-P. Poli, and C. Hudelot. A fuzzy close algorithm for mining fuzzy association rules. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 88–99, 2018.

[68] F. Poursabzi-Sangdeh, D. G. Goldstein, J. M. Hofman, J. W. Vaughan, and H. Wallach. Manipulating and measuring model interpretability. *arXiv preprint arXiv:1802.07810*, 2018.

[69] 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, 2016.

[70] C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[71] E. H. Shortliffe and B. G. Buchanan. A model of inexact reasoning in medicine. *Mathematical Biosciences*, 23(3):351 – 379, 1975.

[72] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013.

[73] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[74] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3319–3328, 2017.

[75] P. Turney. Bias and the quantification of stability. *Machine Learning*, 20(1-2):23–33, 1995.

[76] A. V. Tuzikov, O. Colliot, and I. Bloch. Evaluation of the symmetry plane in 3d mr brain images. *Pattern Recognition Letters*, 24(14):2219–2233, 2003.

[77] M. C. Vanegas, I. Bloch, and J. Inglada. Fuzzy constraint satisfaction problem for model-based image interpretation. *Fuzzy Sets and Systems*, 286:1 – 29, 2016.

[78] D. Waltz. Understanding line drawings of scenes with shadows. In *The Psychology of Computer Vision*, page pages, 1975.

[79] R. R. Yager. The representation of fuzzy relational production rules. *Applied Intelligence*, 1(1):35–42, 1991.

[80] R. R. Yager and D. P. Filev. Relational partitioning of fuzzy rules. *Fuzzy sets and systems*, 80(1):57–69, 1996.

[81] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

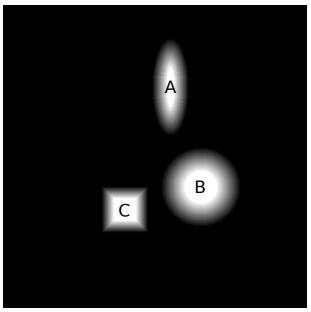
[82] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338 – 353, 1965.

[83] L. A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning—i. *Information sciences*, 8(3):199–249, 1975.

[84] L. A. Zadeh. Fuzzy logic= computing with words. In *Computing with Words in Information/Intelligent Systems 1*, pages 3–23. Springer, 1999.

[85] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833, 2014.

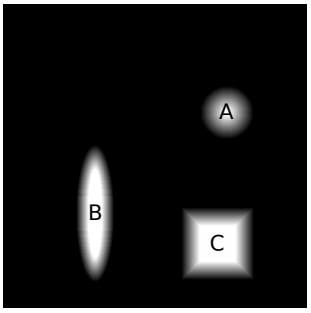
A Toy Dataset: Additional Results



This instance belongs to class 2 with a high confidence because:

- object B *is disk*,
- object C *is square*,
- object A *is ellipse*,
- object C *is to the left of* object B,
- object C *is below* object A,
- object B *is to the right of* object C,
- object B *is below* object A,
- object A *is above* object C,
- object A *is to above* object B.

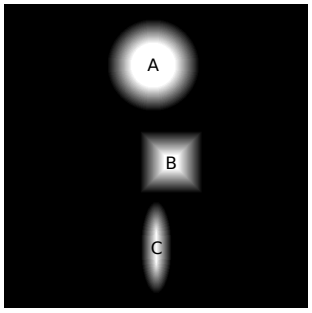
Figure 17: Example of explanation for an instance from class 2. This is one of the borderline example we generated.



This instance belongs to class 3 with a very high confidence because:

- object A *is disk*,
- object C *is square*,
- object B *is ellipse*,
- object C *is below* object A,
- object C *is to the right of* object B,
- object A *is above* object C,
- object A *is to the right of* object B,
- object B *is to the left of* object A,
- object B *is to the left of* object C.

Figure 18: Example of explanation for an instance from class 3.



This instance belongs to class 4 with an average confidence because:

- object A *is disk*,
- object B *is square*,
- object C *is ellipse*,
- object B *is to the right of* object A,
- object B *is above* object C,
- object A *is to the left of* object B,
- object A *is above* object C,
- object C *is below* object A,
- object C *is to below* object B.

Figure 19: Example of explanation for an instance from class 4. This is one of the borderline example we generated.

B Medical Dataset: Additional Results

In this section, we specify the constraints that enabled to generate the explanations in Figure 15 on page 26.

$$
\begin{aligned}
C = \{&(x_{\text{l\_lung}}, x_{\text{r\_lung}}, \mathcal{R}_{\text{completely to the left of}}),\\
&(x_{\text{l\_kidney}}, x_{\text{spleen}}, \mathcal{R}_{\text{below}}),\\
&(x_{\text{r\_psoas}}, x_{\text{r\_kidney}}, \mathcal{R}_{\text{below}}),\\
&(x_{\text{l\_kidney}}, x_{\text{r\_kidney}}, \mathcal{R}_{\text{to the left of}}),\\
&(x_{\text{l\_kidney}}, x_{\text{l\_lung}}, \mathcal{R}_{\text{completely below}}),\\
&(x_{\text{r\_kidney}}, x_{\text{r\_lung}}, \mathcal{R}_{\text{completely below}}),\\
&(x_{\text{l\_lung}}, x_{\text{l\_psoas}}, \mathcal{R}_{\text{above}}),\\
&(x_{\text{spleen}}, x_{\text{r\_lung}}, \mathcal{R}_{\text{to the left of}}),\\
&(x_{\text{liver}}, x_{\text{spleen}}, \mathcal{R}_{\text{completely to the right of}}),\\
&(x_{\text{l\_psoas}}, x_{\text{r\_psoas}}, \mathcal{R}_{\text{completely to the left of}}),\\
&(x_{\text{l\_lung}}, x_{\text{spleen}}, \mathcal{R}_{\text{above}}),\\
&, (x_{\text{r\_kidney}}, x_{\text{spleen}}, \mathcal{R}_{\text{to the right of}}),\\
&(x_{\text{r\_psoas}}, x_{\text{r\_lung}}, \mathcal{R}_{\text{completely below}}),\\
&(x_{\text{spleen}}, x_{\text{r\_kidney}}, \mathcal{R}_{\text{to the left of}}),\\
&(x_{\text{spleen}}, x_{\text{l\_lung}}, \mathcal{R}_{\text{completely below}}),\\
&(x_{\text{r\_lung}}, x_{\text{l\_lung}}, \mathcal{R}_{\text{completely to the right of}}),\\
&(x_{\text{r\_kidney}}, x_{\text{liver}}, \mathcal{R}_{\text{completely below}}),\\
&(x_{\text{r\_lung}}, x_{\text{r\_psoas}}, \mathcal{R}_{\text{above}}),\\
&(x_{\text{l\_psoas}}, x_{\text{l\_lung}}, \mathcal{R}_{\text{completely below}}),\\
&(x_{\text{r\_psoas}}, x_{\text{liver}}, \mathcal{R}_{\text{below}}),\\
&(x_{\text{bladder}}, x_{\text{r\_lung}}, \mathcal{R}_{\text{below}}),\\
&(x_{\text{liver}}, x_{\text{r\_lung}}, \mathcal{R}_{\text{below}}),\\
&(x_{\text{r\_psoas}}, x_{\text{l\_psoas}}, \mathcal{R}_{\text{completely to the right of}}),\\
&(x_{\text{r\_kidney}}, x_{\text{l\_psoas}}, \mathcal{R}_{\text{to the right of}}),\\
&(x_{\text{spleen}}, x_{\text{liver}}, \mathcal{R}_{\text{completely to the left of}}),\\
&(x_{\text{l\_psoas}}, x_{\text{spleen}}, \mathcal{R}_{\text{below}}),\\
&(x_{\text{l\_kidney}}, x_{\text{l\_psoas}}, \mathcal{R}_{\text{above}})\}
\end{aligned}
$$