https://doi.org/10.31449/inf.v49i19.9947 Informatica **49** (2025) 307–336 **307**

Model-Agnostic Explainability for Multi-Label Classification via Formal Concept Analysis

Hakim Radja[1], Yassine Djouadi[2], and Karim Tabia[3]
[1]Computer Science Department, BP 17, RP, Tizi-Ouzou, Mouloud Mammeri University of Tizi-Ouzou, Algeria
[2]RIIMA Laboratory, University of Algiers 1, Algeria
[3]CNRS, CRIL UMR 8188, Université d'Artois, Lens, France.
E-mail: hakim.radja@ummto.dz, y.djouadi@univ-alger.dz, tabia@cril.fr

Keywords: Explainable AI, formal concept analysis (FCA), multi-label classification

Received: July 1, 2025

Multi-label classification refers to a supervised learning problem where a single data instance can correspond to several labels simultaneously. While these models achieve high prediction accuracy, they share some limitations with single-label classifiers, particularly in terms of interpretability and explainability. In this work, we introduce a model-agnostic explainability method that enhances the interpretability of multi-label classification models using Formal Concept Analysis (FCA). Our approach aims to provide users with clearer insights into how these models make predictions, helping them better understand the rationale behind the decisions. Specifically, we address key questions such as: What is the smallest set of features needed for the multi-label classifier f to make a prediction? and Which features are relevant to a specific prediction? By answering these questions, our method seeks to improve user confidence in the model's decisions and foster a deeper understanding of multi-label classification. We introduce the key concepts of the Decisive Attribute Set (DAS) and the Significant Attribute Set (SSA). A DAS is the smallest set of features that can independently lead to a prediction, while an SSA includes all the features that influence the prediction of one or more labels. Additionally, we introduce a dedicated class-specific importance score that quantifies the role of each attribute, based on its frequency and specificity across formal concepts. Using these concepts, we generate clear, interpretable patterns in the form of rules, referred to as "DAS-rules", which provide straightforward explanations for individual predictions. Our method achieves high local fidelity, generates compact explanations, and successfully captures feature interactions, as demonstrated through extensive experiments on three benchmark datasets (Stack Overflow, Yelp, and TMC2007-500). These results demonstrate the novelty and practical effectiveness of our FCA-based framework for multi-label explainability.

Povzetek: Prispevek predstavlja modelno neodvisno metodo razložljivosti za večoznankovno klasifikacijo, ki z uporabo formalne analize konceptov omogoča jasnejše razlage napovedi z identifikacijo ključnih in pomembnih značilk ter generiranjem razumljivih pravil.

1 Introduction

In machine learning (ML), the primary goal of predictive models is often to achieve high accuracy. However, as model complexity increases especially with architectures like deep learning, a major drawback emerges: opacity. These models are frequently described as "black boxes," meaning their decision-making processes remain hidden from users. This lack of transparency presents a significant challenge, particularly in sensitive applications where understanding the reasoning behind a prediction is crucial. Explainability is essential because it allows users to interpret, verify, and comprehend how a model reaches its decisions. Without such insights, it becomes difficult to trust or validate the model's outcomes, leaving users uncertain about the factors driving specific decisions [9].

As noted by Golovin [10], any sufficiently complex system tends to behave like a black box when its ease of use surpasses the effort required to fully understand it. In fact, many of today's top-performing machine learning models, including deep neural networks, fall into this category [11, 18]. While these models exhibit impressive accuracy, their lack of interpretability limits their broader adoption, especially in high-stakes fields like healthcare, finance, and autonomous systems. As machine learning models become increasingly integral to decision-making in these sectors, users are left grappling with crucial questions: Can we trust these predictions? How can we meaningfully interpret the decisions made by these models? The black-box nature of these systems not only erodes trust but also poses a significant barrier to their widespread application. This challenge has led to the rise of Explainable AI (XAI) and interpretable machine learning, fields dedicated to improving model explainability and fostering user confidence [12].

Our work adds to the expanding body of research on explainability by focusing specifically on multi-label classification models, an area that has received comparatively less attention. Unlike single-label classifiers, which predict only one label per data sample, multi-label models can assign multiple labels to each instance, introducing an additional layer of complexity in their interpretation. To tackle this challenge, we leverage Formal Concept Analysis (FCA), a powerful mathematical framework that enables us to break down and explain the predictions made by multi-label classifiers. At the heart of our approach are two novel concepts aimed at addressing key questions regarding the role of features in predictions:

- **Decisive Attribute Set ($\mathcal{DAS}$)**: This represents the minimal subset of features required for the model to make a specific prediction. A $\mathcal{DAS}$ identifies the fewest features that, once known, are sufficient to prompt the classifier to predict a particular label or a sub-set of labels. Unlike traditional feature relevance methods, which rank features by importance but do not ensure their sufficiency for a specific prediction, this approach guarantees that the identified features are sufficient for that prediction.
- **Significant Attribute Set ($\mathcal{SSA}$)**: This includes all the features that appear in at least one Decisive Attribute Set. The intuition behind $\mathcal{SSA}$ is that, while not every feature is decisive on its own, these attributes collectively contribute meaningfully to the model's overall predictive process.
- Building on these two concepts, we introduce an *Attribute importance measure* that quantifies the contribution of each feature to the classifier's predictions. This measure reflects both the frequency and specificity with which a given attribute is involved in the sufficient conditions for predicting a particular label. By assigning a numerical score to each attribute with respect to each class, this metric complements the symbolic explanations provided by $\mathcal{DAS}$ and $\mathcal{SSA}$, offering a richer and more comprehensive understanding of the model's decision-making process.

In addition, we propose explanation rules derived from the $\mathcal{DAS}$ to explain individual predictions, similar to anchor rules [41]. These rules are complemented by numerical scores, which help identify and filter the most commonly occurring rules to explain a prediction. Our methodology is model-agnostic, meaning it can be applied to explain the predictions of any black-box model.

The paper introduces several key contributions in the context of explainable multi-label classification using Formal Concept Analysis (FCA). The main contributions of the paper are the following:

1. The paper defines two novel concepts Decisive Attribute Sets $\mathcal{DAS}$ and Significant Attribute Sets $\mathcal{SSA}$ and relates them to two key types of explanations in multi-label classification.
2. Model-Agnostic Framework for Explainability: The proposed method is designed to work with any black-box multi-label classification model.
3. Flattening and Use of FCA for Explanations: The paper develops a novel way to transform multi-label data into a formal context (a binary relation) that facilitates the application of FCA using off-the-shelf FCA tools. The transformation involves converting a three-dimensional (3D) relationship (instances, features, and labels) into a two-dimensional (2D) context.
4. Reduction Algorithm for Formal Concepts: A reduction algorithm is proposed to reduce the number of formal concepts derived during analysis. This optimization balances computational efficiency with the relevance of explanations, making the approach scalable.
5. Experimental Validation: The approach is evaluated using case studies and experiments on multi-label datasets like Stack Overflow, Yelp, and Aviation Safety Reporting System. These experiments validate the utility of the approach in extracting meaningful explanations and understanding classifier behaviors.

These contributions advance the field of explainable AI by providing a structured, mathematically grounded and model-agnostic method for enhancing explainability in multi-label classifiers.
The structure of the paper is as follows: Section 2 briefly presents multi-label classification and the methodologies used to address its challenges. Section 3 presents a detailed review of existing approaches in explainable AI. In Section 4, we discuss the fundamentals of formal concept analysis. Section 5 and Section 6 outlines our proposed methods and algorithms. Section 7 focuses on empirical evaluations and provides an in-depth analysis of the findings. Section 8 highlights the novelty of our approach and compares it with existing explainability techniques in the literature. Finally, Section 9 summarizes the key contributions of this study and provides concluding insights.

2 Multi-label classification

Multi-label classification assigns multiple labels to a single data instance, extending the traditional single-label classification. This approach is widely used in applications like text categorization [19], image classification [20], and bioinformatics [21]. Formally, a data instance $x = (a_1, a_2, \ldots, a_n)$, where each a_i is a binary attribute, is assigned a subset of labels $y \in 2^{\mathcal{C}}$, given a label set $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$.

Example 1. *Table 1 illustrates predictions from a multi-label classifier for 6 instances described by three attributes (a_1, a_2, a_3, also denoted as $\mathcal{P}$ in FCA notation). The label*

space $\mathcal{C}$ includes three labels (c_1, c_2, c_3). For instance, x_1 is predicted to belong to c_1 and c_3.

Table 1: Example of multi-label classifier predictions

$\mathcal{O}$	$\mathcal{P}$			$\mathcal{C}$		
	a_1	a_2	a_3	c_1	c_2	c_3
x_1	1	1	0	1	0	1
x_2	1	0	0	1	0	0
x_3	1	0	1	1	0	0
x_4	0	0	1	0	1	0
x_5	1	1	1	1	0	1
x_6	1	1	0	1	0	1

Multi-label classification [57, 42, 34] has gained significant attention, leading to the development of various algorithms categorized into three types: problem transformation methods, algorithm adaptation methods, and ensemble methods.

1. **Problem Transformation Methods:** These methods convert multi-label classification problems into single-label equivalents. Prominent techniques include: Binary Relevance, Label Powerset and Label Ranking

2. **Algorithm Adaptation Methods:** These approaches modify existing algorithms to handle multi-label problems. Examples include multi-label decision trees, and adaptations of AdaBoost and neural networks for multi-label scenarios.

3. **Ensemble Methods:** These methods combine problem transformation and algorithm adaptation techniques to improve classification performance. An example is Random k-label (RAKEL), which constructs classifiers using random subsets of labels and combines their predictions through a voting mechanism.

Although machine learning algorithms appear powerful in terms of prediction accuracy, they have their own limitations. One of these is opacity or the lack of transparency, which by nature characterizes *black-box systems*. This means that the internal logic of these systems and the internal workings are hidden from the user, which is a serious disadvantage because it does not allow a human (expert or non-expert) to verify, interpret and understand the reasoning of the system and the way particular decisions are made [9]. In other words, any sufficiently complex system acts as a black-box when it becomes easier to run than to understand [10]. Many machine learning models, including the best-performing models in various fields, belong to this group of black-box systems [11] such as deep artificial neural networks [18].This has led to the emergence of the domain of interpretable and explainable machine learning.

3 Explainable AI

Can we always trust a high-performing model? Such models often make critical decisions, such as approving or rejecting mortgage applications or diagnosing severe medical conditions, where the stakes are high. Even when predictions are accurate, they demand clear explanations. A model that delivers both accurate predictions and understandable, relevant explanations holds significantly greater value. The growing use of machine learning in critical domains has generated significant interest in improving model explainability. Explainable ML aims to ensure transparency and interpretability [1], detect biases [2, 3], assess the safe and reliable operation of models [4, 5, 6], offer clear explanations for the underlying causal mechanisms [7], and provide tools for model improvement and debugging [8]. Existing explainability methods are typically categorized as follows [46]:

- **Intrinsic vs Post-hoc**: Intrinsic methods focus on constructing models that are inherently interpretable, whereas post-hoc methods generate explanations for pre-trained black-box models.

- **Local vs Global**: Local methods explain specific individual predictions, while global methods aim to provide insights into the overall behavior of the model.

- **Agnostic vs Specific**: Agnostic methods are model-independent and can be used across various models, while specific methods are designed exclusively for certain model architectures.

The primary focus in explainable ML research has been on how to provide explanations (for an overview, see [12]). Common techniques for explanation include ranking feature contributions [13], generating prototypes [14], and using decision trees [15] or decision rules [16]. Popular methods like LIME [17] and SHAP [18] are widely used to explain predictions. A crucial element of explainability in machine learning is assessing the importance of the features involved in making predictions. Saarela and Jauhiainen [27] provide an overview of explainability methods focused on feature importance. These methods range from providing basic insights to analyzing more complex models. Generally, simpler models, like linear classifiers, are easier to interpret [28, 29], whereas non-linear models pose greater challenges [30, 31]. The importance of attributes differs across models. For instance, an attribute that is critical for one model may be insignificant for another [32]. Turini et al. [12] categorize different approaches that link explainability to specific types of black-box models. Darwiche [33] introduces concepts such as "sufficient" and "necessary" reasons, which help define the minimal set of attribute values required for a prediction, categorizing attributes into these sets according to their importance. Comprehensive surveys further discuss the diversity of explainability techniques and their underlying assumptions [53, 54, 55].

Several recent studies have focused on explainability for multi-label classification. Ciaperoni et al. [49] propose a symbolic multi-label classification model that outputs a small set of interpretable IF-THEN rules, offering a balance between predictive performance and human readability. Ayad et al. [50] extend Shapley values to classifier chains in a multi-label setting, capturing both direct and indirect feature contributions through their *Shapley Chains* method. Hemal and Saha [51] propose an explainable deep learning-based meta-classifier tailored for the multi-label classification of retinal diseases, integrating interpretation at the output level for each predicted label. Mylonas et al. [52] adapt local explanation techniques to random forest classifiers in a multi-label context, generating label-wise explanations for individual instances. However, these approaches are often tied to specific model architectures and deliver only particular types of explanations, which limits their general applicability across different classifiers and explanation needs.

The challenge in explaining black-box models lies in generating interpretable outputs that clarify a specific prediction without fully revealing the model's internal logic. This paper focuses on symbolic explanations, especially those that provide "sufficient reasons" to justify predictions. These explanations identify the factors that are necessary or sufficient for a particular prediction to occur. Our approach is grounded in formal concept analysis, presented in the next section.

4 Formal concept analysis

Formal Concept Analysis (FCA) [22], introduced by Wille in 1982, is an algebra-based approach that facilitates the discovery of potentially useful abstractions. It is applied in various fields:

- **In data mining**: To identify groups of products frequently purchased together, such as in consumer basket analysis.
- **In information retrieval**: To exploit an analogy between the object-property relationship and the document-term relationship for query refinement.
- **In knowledge representation**: To build clusters of knowledge known as *formal concepts*.

This method is based on a **formal context** $K = (O, P, R)$, where:

- O is the set of objects,
- P is the set of properties,
- $R \subseteq O \times P$ is a Boolean relation between certain objects in O and certain properties in P.

Formal concepts are induced using a set-theoretic operator called the **Galois derivation operator** (Δ):

R	hunt	2legs	fly	4legs
lion	X			X
duck		X	X	
eagle	X	X	X	
cow				X

Figure 1: Illustration of a formal context

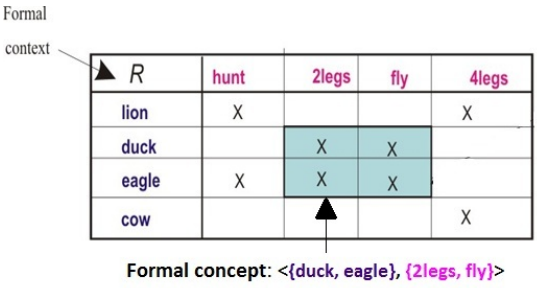

Figure 2: An example of a formal concept

- For a subset $X \subseteq O$, X^Δ represents the properties shared by all objects in X:

$$X^\Delta = \{p \in P \mid \forall o \in X,\ (o, p) \in R\}.$$

  Example: $\{DUCK, EAGLE\}^\Delta = \{2legs, fly\}$.

- For a subset $Y \subseteq P$, Y^Δ represents the objects sharing all properties in Y:

$$Y^\Delta = \{o \in O \mid \forall p \in Y,\ (o, p) \in R\}.$$

  Example: $\{2legs, fly\}^\Delta = \{DUCK, EAGLE\}$.

A **formal concept** is a pair $\langle X, Y \rangle$ such that $X^\Delta = Y$ and $Y^\Delta = X$.

- X is the *extent* of the concept (the maximal set of objects).
- Y is the *intent* of the concept (the maximal set of properties).

Example: $\langle \{DUCK, EAGLE\}, \{2legs, fly\} \rangle$ is a formal concept where $\{DUCK, EAGLE\}$ is the **extent**, and $\{2legs, fly\}$ is the **intent**.

The set of all formal concepts obtained is organized into a **complete lattice**, based on a partial order $\leq$, defined as follows:

$$\langle X_1, Y_1 \rangle \leq \langle X_2, Y_2 \rangle \iff X_1 \subseteq X_2 \text{ (equivalently, } Y_2 \subseteq Y_1)$$

Formal Concept Analysis (FCA) offers several advantages for data analysis. One of its key benefits is the visual representation of the concept lattice, which facilitates an intuitive exploration of the data and its underlying structure. In multi-label classification, FCA has been

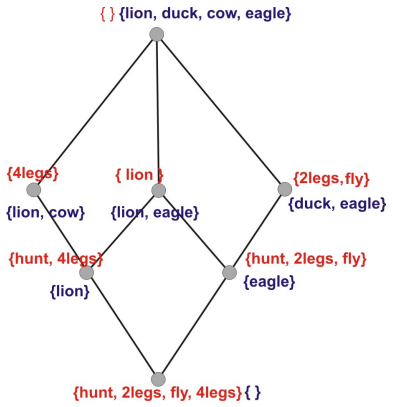

Figure 3: The concept lattice corresponding to the formal context of Figure 2

widely studied to enhance the performance of classification models. Research has demonstrated that FCA effectively addresses challenges specific to multi-label classification, such as handling noisy labels, reducing dimensionality, and learning from limited datasets. Li et al. [37] proposed a framework based on Formal Concept Analysis (FCA) for addressing multi-label classification, leveraging the conceptual structure of the data and achieving promising results in terms of precision and recall. Zhang et al. [38] proposed an innovative approach that combines FCA with deep learning techniques to capture highly discriminative features in multi-label data. Wang et al. [39] extended FCA to model label relationships, thereby improving the representation of complex class interactions. Additionally, Chen et al. [40] explored the use of fuzzy FCA to manage uncertainty and ambiguity in data annotations.

FCA is also used in the area of explainability of machine learning models. For example, in [43] the authors explore the application of Shapley values, derived from cooperative game theory, to assess the contribution of individual attributes in classification and clustering based on formal concepts. In [44] the authors explore how FCA can offer techniques to provide plausible explanations for model predictions, focusing on the importance of individual attributes in classification and clustering. Similarly, in [45], the authors propose a framework based on FCA to explain deep learning models that are considered as ”black boxes”. The authors apply FCA to analyze anomalies in data, which is then used to explain the results of machine learning models, focusing on binary classification tasks.

Building on this background, the next section outlines our approach to explaining multi-label classification using FCA.

5 FCA Representation for Multi-Label Classification Predictions

Explanations aim to clarify how specific input features influence a model's output, fostering trust and enabling users to validate its decisions.

In our approach, explanations are based on *decisive attributes* and *formal concepts*, which are used to identify the key features that drive a prediction.

Our approach to explaining the predictions of a multi-label classification model starts by associating a formal context with the model's prediction(s). Specifically, we use a standard formal context to represent instances, attributes, and labels. This raises an important question: why choose traditional Formal Concept Analysis (FCA) over triadic[1] FCA for this task? Classical FCA focuses on binary relationships between objects and attributes, making it algorithmically simpler and computationally more efficient than triadic FCA. In contrast, triadic FCA provides greater expressiveness by extending the representation to ternary relationships involving objects, attributes, and attribute values. However, this added expressiveness comes at the cost of increased computational complexity, even with smaller datasets [35] [36]. Classical FCA strikes a better balance between expressiveness and computational feasibility, particularly when applied to large-scale datasets.

We now provide an overview of our FCA-based explainability approach:

FCA-based explanation process: step-by-step overview

To provide a clearer understanding of the thinking behind our approach, we provide a concise five-step overview of how symbolic explanations are derived from a multi-label classifier using Formal Concept Analysis (FCA). This illustration is complemented by a simplified diagram (Figure 4) and a simplified example for illustration purposes.

Step 1: Classifier predictions We consider a dataset with three instances (x_1, x_2, x_3), three binary attributes (a_1, a_2, a_3), and two class labels (c_1, c_2). The predictions of a multi-label classifier f are as follows:

Object	a_1	a_2	a_3	c_1	c_2
x_1	1	0	1	1	0
x_2	0	1	0	1	1
x_3	1	1	1	1	1

So we have: $f(x_1) = \{c_1\}$, $f(x_2) = \{c_1, c_2\}$, $f(x_3) = \{c_1, c_2\}$.

[1] A triadic context consists of sets objects $\mathcal{O}$, set of attributes $\mathcal{P}$ and a set of conditions, $\mathcal{C}$. It extends binary formal contexts to represent ternary relations encoding under which condition c and an object o have attribute a.

Step 2: Ternary relation construction Each instance is represented as a set of active features and predicted classes. We build a ternary relation $R \subseteq O \times P \times C$, where a triple (x_i, a_j, c_k) indicates that attribute a_j is active in instance x_i and contributes to the prediction of class c_k.

- $x_1 \rightarrow (x_1, a_1, c_1), (x_1, a_3, c_1)$
- $x_2 \rightarrow (x_2, a_2, c_1), (x_2, a_2, c_2)$
- $x_3 \rightarrow$ all six possible triplets: $(x_3, a_1, c_1), (x_3, a_2, c_1), (x_3, a_3, c_1),$ $(x_3, a_1, c_2), (x_3, a_2, c_2), (x_3, a_3, c_2)$

To apply FCA, we flatten each attribute-class pair into a compound attribute (c_k, a_j). The resulting binary context is:

Object	$(c_1 - a_1)$	$(c_1 - a_2)$	$(c_1 - a_3)$	$(c_2 - a_1)$	$(c_2 - a_2)$	$(c_2 - a_3)$
x_1	1	0	1	0	0	0
x_2	0	1	0	0	1	0
x_3	1	1	1	1	1	1

This transformation is crucial since it allows us to encode the ternary relation into a standard two-dimensional binary context. In this way, we can directly leverage existing Formal Concept Analysis (FCA) tools, which are inherently designed to operate on binary contexts defined in two dimensions.

Step 3: Formal concept extraction We compute formal concepts $\langle X, Y \rangle$ from the binary context, where X is a set of objects and Y a set of shared compound attributes. For example:

$$\langle \{x_1, x_3\}, \{(c_1 - a_1), (c_1 - a_3)\} \rangle$$

indicates that both x_1 and x_3 share these two conditions for predicting class c_1.

Step 4: Reducing redundant concepts In real-world applications, the number of extracted formal concepts can be very large, often with significant redundancy. To enhance interpretability, we apply a reduction step that retains only the most relevant concepts, ensuring that the resulting explanations are concise and non-repetitive.

Step 5: Deriving explanations From the reduced set of formal concepts obtained, we derive symbolic explanations that describe how features contribute to the predicted labels. The different types of explanations and their computation are detailed in the following sections.

The following figure summarizes the different steps of our FCA-based approach.

The classifier's predictions are transformed into a formal context using a flattening step. Formal concepts are then computed, from which symbolic explanations are derived.

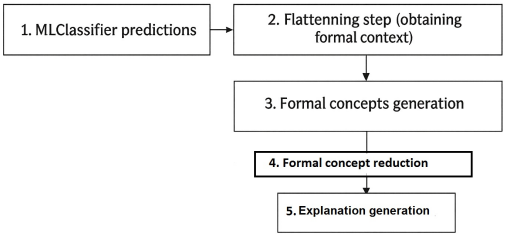

Figure 4: A step-by-step overview of the FCA-based explanation process

From illustrative example to formal framework

In the following, we provide a formal and detailed description of the methodology outlined above. While the previous illustration was based on a simplified example, the next sections generalize the approach and define its components more rigorously. The five steps described earlier are synthesized into three main stages:

1. **Constructing a formal representation to encode classifier outputs**
   In multi-label classification, a classifier f assigns each data instance x, characterized by a feature vector $(a_1, a_2, \ldots, a_n)$, to a subset of labels y from the set $\mathcal{C} = \{c_1, c_2, \ldots, c_k\}$. This relationship can be represented as a 3D structure, including objects $\mathcal{O}$ (data instances), attributes $\mathcal{P}$, and labels $\mathcal{C}$, mathematically expressed as $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{P} \times \mathcal{C}$. Converting this 3D structure into a 2D formal context is done in such a way that no information is lost.

2. **Deriving explanations through formal concepts**
   By transforming the 3D context into a 2D structure, formal concepts are generated, grouping objects with identical class predictions under the same conditions. To manage the potentially large number of concepts, a filtering process retains only the most relevant ones.

3. **Generating explanations**
   In this step, we present the various types of explanations derived from formal concepts. The *Decisive Attribute Set* ($\mathcal{DAS}$) is the smallest subset of features that allows the model f to predict a specific class for an instance, independently of the other attribute values. From the $\mathcal{DAS}$, we can derive other explanations, such as the *Significant Attributes Set* ($\mathcal{SSA}$), *necessary attributes set* ($\mathcal{NAS}$), and $\mathcal{DAS}$-rules.

The next sections detail the three stages: constructing the ternary relation, transforming it into a formal context, and extracting symbolic explanations.

5.1 Formal representation of classifier outputs

The process of generating explanations for multi-label classifier predictions begins with converting a multi-label dataset into a formal context. To start, let's define the key components of this transformation.

Let $\mathcal{MLD} = (\mathcal{O}, \mathcal{P}, \mathcal{C}, \mathcal{R})$ be a multi-label dataset, where $\mathcal{O}$ is a set of instances (or objects), $\mathcal{P}$ is a set of features (or attributes), $\mathcal{C}$ is a set of classes, and $\mathcal{R}$ is a ternary relation $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{P} \times \mathcal{C}$, representing the relations between the instances, their features, and their associated class predictions. The goal is to transform this multi-label data, represented as a ternary relation $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{P} \times \mathcal{C}$, into a binary relation $\mathcal{R}' \subseteq \mathcal{O} \times \mathcal{M}$, where $\mathcal{M}$ represents pairs of features and classes. This transformation, also called "flattening" of the data, is similar to what is done in [24]. As a result, a formal context $\mathcal{K}' = (\mathcal{O}, \mathcal{M}, \mathcal{R}')$ is created, where $\mathcal{O}$ represents the set of objects (instances), $\mathcal{M} = (\mathcal{C} \times \mathcal{P})$ is a set of pairs (c_k, a_j), where $a_j \in \mathcal{P}$ represents a feature, and $c_k \in \mathcal{C}$ represents a class, and $\mathcal{R}' \subseteq \mathcal{O} \times \mathcal{M}$ is a binary relation between objects and the set of pairs (c_k, a_j).

In simpler terms, this transformation associates each feature a_j with a class c_k to create a new set of combined attributes, allowing us to represent the original multi-dimensional data in a format compatible with Formal Concept Analysis (FCA). More specifically, for each instance $x_i \in \mathcal{O}$, the classifier f assigns predictions for one or more classes based on the feature values of x_i. The binary relation $\mathcal{R}'$ for an instance x_i and a pair (c_k, a_j) is defined as:

$$\mathcal{R}'(x_i, (a_j \times c_k)) = \begin{cases} 1 & \text{if } \mathcal{R}(x_i, a_j, c_k) = 1 \\ 0 & \text{otherwise} \end{cases}$$

By doing so, the ternary relationship between instances, features, and classes is flattened into a 2D relationship that can be directly processed using existing tools for FCA analysis.

Flattening the formal context is essential for applying FCA to generate explanations. By transforming the 3D data into a 2D binary context, we can identify formal concepts, which are the fundamental structures in FCA representing meaningful relationships between objects and attributes. In this context, a formal concept consists of a set of objects (instances) that share common attributes (feature–class pairs). Once the formal concepts are computed, they can be used to generate different types of explanations. Formal concepts naturally organize the data into clusters where instances display similar behavior, enabling us to derive explanations that are both interpretable and consistent with the classifier's predictions. In our approach, we use the Next Closure algorithm [26] to compute the formal concepts from the flattened formal context. This algorithm computes the closed itemsets based on the lexicographical order of attributes, making it efficient even for large datasets.

Example 2. *Table 2 provides an example of representing multi-label data as a formal context.*

5.2 Deriving explanations through formal concepts

To extract explanations from the formal context, the derivation operator is adapted to suit the context of multi-label classification. For a 2D formal context, the goal is to generate explanations for the classifier's predictions. Let $\mathcal{K}' = (\mathcal{O}, \mathcal{M}, \mathcal{R}')$ be the formal context, defined as follows:

- $\mathcal{O}$ represents the set of data instances,
- $\mathcal{M} = \mathcal{C} \times \mathcal{P}$ is the set of attribute-class pairs (c_k, a_j), where $a_j \in \mathcal{P}$ is an attribute, and $c_k \in \mathcal{C}$ is a class label,
- $(x_i, (c_k, a_j)) \in \mathcal{R}'$ indicates that the instance $x_i \in \mathcal{O}$ is associated with class c_k when the attribute a_j is present.

For any subset $X \subseteq \mathcal{O}$ and $Y \subseteq \mathcal{M}$, with $Y = \{(c_k, a_j) \mid a_j \in \mathcal{P}, c_k \in \mathcal{C}\}$, the Galois derivation operator $(.)^\Delta$ is defined as follows:

$$X^\Delta = \{(c_k, a_j) \in \mathcal{M} \mid X \subseteq \mathcal{R}'(c_k, a_j)\}$$

$$Y^\Delta = \{x \in \mathcal{O} \mid Y \subseteq \mathcal{R}'(x)\}$$

Here:

- X^Δ is the set of attribute-class pairs shared by all instances in X,
- Y^Δ is the set of instances that possess all attribute–class pairs in Y.

In this context, a formal concept is a pair $\langle X, Y \rangle$ such that $X \subseteq \mathcal{O}$, $Y \subseteq \mathcal{M}$, with the properties that $X^\Delta = Y$ and $Y^\Delta = X$. The set of all formal concepts is denoted by $\mathcal{L}(\mathcal{K}')$.

5.2.1 Example

Referring to the formal context from Table 2, the derived formal concepts are:

- $fc_1 : \langle \{x_1, x_2, x_3, x_4, x_5, x_6\}, \{\} \rangle$,
- $fc_2 : \langle \{x_4\}, \{(c_2, a_3)\} \rangle$,
- $fc_3 : \langle \{x_1, x_2, x_3, x_5, x_6\}, \{(c_1, a_1)\} \rangle$,
- $fc_4 : \langle \{x_1, x_5, x_6\}, \{(c_1, a_1), (c_1, a_2), (c_3, a_1), (c_3, a_2)\} \rangle$,
- $fc_5 : \langle \{x_3, x_5\}, \{(c_1, a_1), (c_1, a_3)\} \rangle$,
- $fc_6 : \langle \{x_5\}, \{(c_1, a_1), (c_1, a_2), (c_1, a_3), (c_3, a_1), (c_3, a_2), (c_3, a_3)\} \rangle$,
- $fc_7 : \langle \{\}, \{(c_1, a_1), (c_1, a_2), (c_1, a_3), (c_2, a_1), (c_2, a_2), (c_2, a_3), (c_3, a_1), (c_3, a_2), (c_3, a_3)\} \rangle$.

Table 2: An example of transforming 3D multi-label data into a 2D formal context

$\mathcal{O}$	$\mathcal{P}$			$\mathcal{C}$		
	a_1	a_2	a_3	c_1	c_2	c_3
x_1	1	1	0	1	0	1
x_2	1	0	0	1	0	0
x_3	1	0	1	1	0	0
x_4	0	0	1	0	1	0
x_5	1	1	1	1	0	1
x_6	1	1	0	1	0	1

$\mathcal{O}$	$\mathcal{M} = \mathcal{C} \times \mathcal{P}$								
	c_1-a_1	c_1-a_2	c_1-a_3	c_2-a_1	c_2-a_2	c_2-a_3	c_3-a_1	c_3-a_2	c_3-a_3
x_1	1	1	0	0	0	0	1	1	0
x_2	1	0	0	0	0	0	0	0	0
x_3	1	0	1	0	0	0	0	0	0
x_4	0	0	0	0	0	1	0	0	0
x_5	1	1	1	0	0	0	1	1	1
x_6	1	1	0	0	0	0	1	1	0

For instance, the formal concept fc_4 can be rewritten as:

$$fc_4 : \langle \{x_1, x_5, x_6\}, \{c_1, c_3\}, \{a_1, a_2\} \rangle$$

This means that classes c_1 and c_3 are predicted for the instances x_1, x_5, and x_6 when attributes a_1 and a_2 are present (their value is set to 1). Here, $\{x_1, x_5, x_6\}$ forms the extent, $\{c_1, c_3\}$ forms the intent, and $\{a_1, a_2\}$ represents the condition of the formal concept fc_4.

The flattened formal context, as previously constructed, plays a crucial role in generating these explanations. Each formal concept derived from the context captures information about a set of instances, their predicted classes, and the conditions (attributes) under which these predictions are made. As a result, the formal concept naturally offers an explanation by identifying the attributes that influence the classifier's predictions and grouping instances with similar characteristics and outcomes. Since the formal concepts are derived from the flattened context, they serve as direct representations of explanations. Each concept connects data instances, class predictions, and the relevant attributes, making the Galois lattice of concepts an ideal framework for organizing and interpreting the classifier's decisions.

By proceeding in this way with the set of all formal concepts obtained in the previous example, we obtain the following set of formal concepts:

- $fc_1 : \langle \{x_1, x_2, x_3, x_4, x_5, x_6\}, \{\}, \{\} \rangle$,
- $fc_2 : \langle \{x_4\}, \{c_2\}, \{a_3\} \rangle$,
- $fc_3 : \langle \{x_1, x_2, x_3, x_5, x_6\}, \{c_1\}, \{a_1\} \rangle$,
- $fc_4 : \langle \{x_1, x_5, x_6\}, \{c_1, c_3\}, \{a_1, a_2\} \rangle$,
- $fc_5 : \langle \{x_3, x_5\}, \{c_1\}, \{a_1, a_3\} \rangle$,
- $fc_6 : \langle \{x_5\}, \{c_1, c_3\}, \{a_1, a_2, a_3\} \rangle$,
- fc_7: $\langle \{\}, \{c_1, c_2, c_3\}, \{a_1, a_2, a_3\} \rangle$.

5.2.2 Minimizing formal concepts

The large number of formal concepts can affect their relevance for explanations. Algorithm 1 reduces this number by taking into account attributes, instances, and classes. For instance, in Example 5.2.1, the intent $\{c_1\}$ is found in multiple concepts.

- In fc_3: the class c_1 is predicted for the objects x_1, x_2, x_3, x_4, x_5, and x_6 when the attribute a_1 is present.
- In fc_5: the class c_1 is predicted for the objects x_3 and x_5 when the attributes a_1 and a_3 are present.

It is evident that the attribute a_1 alone suffices to classify objects x_3 and x_5 in class c_1, as $\{x_3, x_5\} \subseteq \{x_1, x_2, x_3, x_4, x_5, x_6\}$ and $\{a_1\} \subseteq \{a_1, a_3\}$. Hence, a_3 is not essential for predicting c_1 for the objects $\{x_3, x_5\}$. From an explainability perspective, a_3 is irrelevant, and retaining only the formal concept fc_3 is adequate. Leveraging this observation, Algorithm 1 aims to minimize the number of formal concepts: Let $\mathcal{L}(K') = \{fc_1, fc_2, \ldots, fc_n\}$ denote the set of all formal concepts, $\mathcal{I}nt(fc_i)$ refer to the intent of the formal concept fc_i, $\mathcal{E}xt(fc_i)$ indicate the extent of the formal concept fc_i, and $\mathcal{C}ond(fc_i)$ represent its condition. The objective is to reduce the number of formal concepts by removing redundant ones based on their intents, extents, and conditions. The main steps of this algorithm are: **Input:** A set of formal concepts $\mathcal{L}(K') = \{fc_1, fc_2, \ldots, fc_n\}$.

Output: A reduced set of formal concepts $\mathcal{L}'(K')$.

Steps:

1. Initialize an empty set for the reduced formal concepts, $\mathcal{L}'(K') = \emptyset$.
2. For each pair of formal concepts fc_i and fc_j:
   - If their intents are identical $\mathcal{I}nt(fc_i) = \mathcal{I}nt(fc_j)$:

- Check if the extent of fc_i is a subset of the extent of fc_j: $\mathcal{E}xt(fc_i) \subseteq \mathcal{E}xt(fc_j)$.
- If true, ensure that the condition of fc_i is a super-set of the condition of fc_j: $\mathcal{C}ond(fc_i) \supseteq \mathcal{C}ond(fc_j)$.
- If all conditions are met, add fc_i to the reduced set, $\mathcal{L}'(K')$.

Algorithm 1 Formal Concepts Reduction

Input: $\mathcal{L}(K') = \{fc_1, \ldots, fc_n\}$
Output: $\mathcal{L}'(K')$ ▷ Reduced set of formal concepts

```
L'(K') ←
for i ← 1 to n − 1 do                    ▷ n is the size of L(K')
    for j ← i + 1 to n do
        if Int(fc_i) = Int(fc_j) then
            if Ext(fc_i) ⊆ Ext(fc_j) then
                if Cond(fc_i) ⊇ Cond(fc_j) then
                    L'(K') ← L'(K') ∪ {fc_i}    ▷ Add fc_i to L'(K')
```

5.2.3 Example (example 5.2.1 continued)

By applying the reduction algorithm to the set $\mathcal{L}(K')$ derived in Example 5.2.1, the reduced set of formal concepts is obtained as $\mathcal{L}'(K') = \{fc_1, fc_2, fc_3, fc_4, fc_7\}$.

The time complexity of the algorithm is $O(n^2)$, where n is the size of the input set $\mathcal{L}(K')$. This quadratic complexity is manageable for moderately-sized datasets, allowing for efficient reduction while ensuring the exactness of explanations.

So far, we have demonstrated the construction of a formal context for explanations and the reduction of formal concepts. The next section focuses on generating explanations from the resulting formal concepts.

6 Generating explanations

Our method for explaining multi-label classification operates in a model-agnostic manner and delivers symbolic explanations in the form of sufficient reasons.

In this section, we further elaborate on the example presented in Table 2. As detailed in the previous section, the reduced set of formal concepts, $\mathcal{L}'(K')$, obtained is:

- $fc_1 : \langle\{x_1, x_2, x_3, x_4, x_5, x_6\}, \{\}, \{\}\rangle$,
- $fc_2 : \langle\{x_4\}, \{c_2\}, \{a_3\}\rangle$,
- $fc_3 : \langle\{x_1, x_2, x_3, x_5, x_6\}, \{c_1\}, \{a_1\}\rangle$,
- $fc_4 : \langle\{x_1, x_5, x_6\}, \{c_1, c_3\}, \{a_1, a_2\}\rangle$,
- fc_7: $\langle\{\}, \{c_1, c_2, c_3\}, \{a_1, a_2, a_3\}\rangle$.

fc_1 and fc_7 can be removed from $\mathcal{L}'(K')$ as they do not provide any useful information, resulting in the new set of formal concepts $\mathcal{L}'(K')$:

- $fc_2 : \langle\{x_4\}, \{c_2\}, \{a_3\}\rangle$,
- $fc_3 : \langle\{x_1, x_2, x_3, x_5, x_6\}, \{c_1\}, \{a_1\}\rangle$,
- $fc_4 : \langle\{x_1, x_5, x_6\}, \{c_1, c_3\}, \{a_1, a_2\}\rangle$,

For any data instance x with prediction $f(x)$, our approach provides symbolic explanations in the form of *sufficient reasons*. These explanations consist of subsets of features from x that justify its prediction $y = f(x)$.

6.1 Sufficient reason explanations

In the context of explainable AI, we aim to find explanations that are concise, as overly long explanations can become difficult to interpret. Therefore, the goal is to identify the smallest possible explanation that still effectively justifies a prediction. In our case, this means identifying the smallest subset of attributes that, once determined, triggers the prediction regardless of the values of the remaining attributes. A sufficient reason consists of a sub-set of features that, when present (namely, their values set to 1), provide a sufficient explanation for classifying a data instance into a specific class. For instance, in a medical diagnosis model, features such as Fever and cough may provide a sufficient reason for classifying a patient as having a respiratory infection. In our approach, sufficient reason explanations are represented by the concept of the *Decisive Attribute Set*.

Definition 1. (*Decisive Attribute Set*)
The smallest (in the sense of set inclusion) set of attributes enabling the classification of a data instance into a given class is referred to as the '*Decisive Attribute Set ($\mathcal{DAS}$)*.'

In FCA terms, a sufficient reason corresponds to the concept of a *Decisive Attribute Set ($\mathcal{DAS}$)*. Formally, a $\mathcal{DAS}$ is defined as the smallest subset of features $\{a_j \in \mathcal{P}\}$ such that a classifier f predicts $y \subseteq \mathcal{C}$ for an instance $x \in \mathcal{O}$, independently of the values of the remaining attributes. This minimality ensures that the prediction remains valid regardless of variations in the unused attributes.

Let $x = (a_1, a_2, \ldots, a_n)$ represent the data instance to classify, and let $c_k \in y \subseteq \mathcal{C}$ be a class included in the multi-label prediction $y = f(x)$, where f is the multi-label classifier to be explained.

We recall that in the context of multi-label classification, a formal concept is defined as a triple:

$$\langle \text{Extent, Intent, Condition} \rangle$$

where:

- the **Extent** (**Ext**(fc_i)) is the set of instances for which the concept holds,
- the **Intent** (**Int**(fc_i)) is the set of predicted class labels for those instances,
- the **Condition** (**Cond**(fc_i)) is the set of input attributes that characterize the instances and explain the predicted labels.

For example, a concept of the form $\langle\{x_1, x_5, x_6\}, \{c_1, c_3\}, \{a_1, a_2\}\rangle$ expresses that instances x_1, x_5, and x_6 share the feature condition $\{a_1, a_2\}$, which leads the classifier to assign them the labels c_1 and c_3.

To better understand the relationship between sufficient reasons and decisive attributes, we define $\mathcal{B}(c_k)$ as the set of all formal concepts whose intents contain the class c_k. Let $\mathcal{L}'(K') = \{fc_1, fc_2, \ldots\}$ represent the reduced set of formal concepts derived using Algorithm 1. Specifically, $\mathcal{B}(c_k) = \{fc_i \in \mathcal{L}'(K') \mid c_k \in \mathcal{I}nt(fc_i)\}$.

The $\mathcal{DAS}$ that enables the model f to predict the class c_k is further defined as follows:

- $\mathcal{B}(c_k)$: the set of all formal concepts whose intents include the class c_k,
- $\mathcal{E}xt(fc_i) = \{x_i \in \mathcal{O} \mid fc_i \in \mathcal{B}(c_k)\}$: the extent of the formal concept containing c_k in its intent,
- $\mathcal{C}ond(fc_i) = \{a_i \in \mathcal{P} \mid fc_i \in \mathcal{B}(c_k)\}$: the condition of the formal concept containing c_k in its intent,
- $\mathcal{D}iff(c_k) = \bigcup_{i=1}^{n} \mathcal{E}xt(fc_i) \setminus \bigcap_{i=1}^{n} \mathcal{E}xt(fc_i)$: the set difference between the union and the intersection of the extents of formal concepts having c_k in their intents.

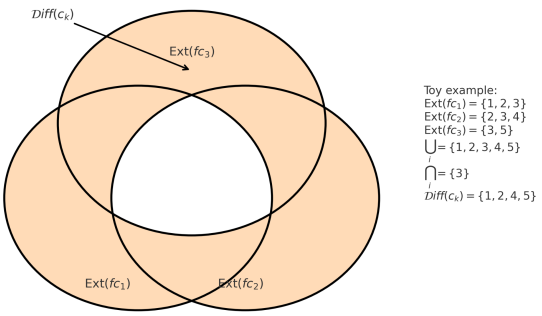

Figure 5: Illustration of $\mathcal{D}iff(c_k)$ with the distinctive instances (colored area) that appear in at least one but not all extents for c_k

Intuition: $\mathcal{D}iff(c_k)$ contains the *distinctive instances* that appear in at least one concept for c_k but not in all of them. This operation removes the instances that appear in all concepts, as they do not provide any variation to explain the prediction. Only instances in $\mathcal{D}iff(c_k)$ help differentiate between different decision contexts, and therefore, they are significant for interpretation. By focusing on these distinctive instances, we capture the diversity of scenarios leading to c_k, making the explanation more specific and informative.

The Decisive Attribute Set ($\mathcal{DAS}$) for the class c_k is computed as:

$$\mathcal{DAS}_j(c_k) = \{\bigcup \mathcal{C}ond(fc_i) | x_j \in \mathcal{E}xt(fc_i) \text{ and } x_j \in \mathcal{D}iff(c_k)\}.$$

Below an algorithm to summarize the procedure for computing the Decisive Attribute Set $\mathcal{DAS}$:

Algorithm 2 Compute $\mathcal{DAS}(c_k)$

Input: $\mathcal{B}(c_k)$
Output: $\mathcal{DAS}(c_k)$

$\mathcal{DAS}(c_k) = \emptyset$
if $|\mathcal{B}(c_k)|$=1 **then**
  $\mathcal{DAS}(c_k) = \mathcal{C}ond(fc_k)$
else
  for $i \leftarrow 1, |\mathcal{B}(c_k)|$ **do**
    $\mathcal{D}iff(c_k) = \bigcup \mathcal{E}xt(fc_i) \setminus \bigcap \mathcal{E}xt(fc_i)$
    for each $x \in \mathcal{D}iff(c_k)$ **do**
      for $i = 1 \leftarrow 1, n$ **do**
        if $x \in \mathcal{E}xt(fc_i)$ **then**
          $\mathcal{DAS}_x(c_k) = \bigcup \mathcal{C}ond(fc_i)$
        if $\mathcal{DAS}_x(c_k) \notin \mathcal{DAS}(c_k)$ **then**
          $\mathcal{DAS}(c_k) = \mathcal{DAS}(c_k) \cup \mathcal{DAS}_x(c_k)$
  ▷ Add a subset to $\mathcal{DAS}(c_k)$

The algorithm ensures completeness through its exhaustive iterative strategy across all relevant formal concepts. At each iteration, the algorithm systematically considers all pertinent formal concepts and evaluates each instance associated with these concepts. This methodical approach guarantees coverage of every conceivable scenario within the conceptual space, affirming the algorithm's completeness in addressing a specific problem.

After we have obtained the set of reduced formal concepts $\mathcal{L}'(K')$, we use it to calculate the Decisive Attribute sets for a given class.

To derive the Decisive Attribute Sets $\mathcal{DAS}(c_k)$ for the class c_k, we follow these steps:

1. **Identify Relevant Formal Concepts:**
   Compute the set $\mathcal{B}(c_k)$ consisting of all formal concepts in $\mathcal{L}'(K')$ whose intents include the class c_k.
2. **Handle Single Formal Concept Case:**
   If $|\mathcal{B}(c_k)| = 1$, this implies that only one formal concept includes c_k in its intent.
   In this case, the Decisive Attribute Set for predicting c_k is simply the condition of this unique formal concept.
   Recall that a formal concept in our approach is represented as $fc : \langle\{extent\}, \{intent\}, \{condition\}\rangle$, meaning:
   - The classes in $\{intent\}$ are predicted for the instances in $\{extent\}$,
   - provided the attributes in $\{condition\}$ are present.
3. **Handle Multiple Formal Concepts Case:**
   If c_k appears in multiple formal concepts, we analyze

the representative instances ($\mathcal{D}iff(c_k)$) of each formal concept.
The representative instances are obtained by calculating the set difference between the union and the intersection of the extents of the formal concepts in $\mathcal{B}(c_k)$
For each instance x in $\mathcal{D}iff(c_k)$, we keep only the formal concepts that have the instance x in their extents. Then, we calculate the $\mathcal{DAS}_x(c_k)$ by making the union of these formal concepts intents.

Intuitively, a sufficient reason for a class c_k is a minimal set of attributes R that fully predicts and explains the predicted class, meaning that the presence of R implies the prediction of c_k, with no superfluous attributes in R. Similarly, a DAS is a set of attributes sufficient to predict c_k, containing only the attributes necessary for that prediction. The minimality of R and DAS ensures that these sets contain only the essential attributes, without redundancy. Thus a DAS is a sufficient reason because it includes the attributes necessary and sufficient for the prediction, and a sufficient reason is a DAS because it is both minimal and complete in justifying the predicted class. This equivalence directly drives the design of Algorithm 2. By systematically identifying the DAS for each class, the algorithm computes the minimal sufficient reasons for the predictions. Its completeness and correctness, as discussed, further ensures that no essential explanatory element is overlooked. Algorithm 2 ensures both completeness and correctness in the computation of Decisive Attribute Sets ($\mathcal{DAS}(c_k)$). Completeness is achieved because the algorithm examines all relevant formal concepts ($\mathcal{B}(c_k)$) for a given class c_k and identifies the minimal set of attributes that are necessary and sufficient to predict that class. Using representative instances ($\mathcal{D}iff(c_k)$) ensures that every relevant case is considered. Correctness arises from the methodical application of formal concept properties: the extents and intents of these concepts are utilized to guarantee that the pertinent attribute sets meet the criteria of minimality and predictiveness.

It is important to highlight that a Decisive Attribute Set ($\mathcal{DAS}$) enabling the prediction of a class c_k is not necessarily unique. Indeed, multiple Decisive Attribute Sets can achieve the same prediction. The collection $\mathcal{DAS}(c_k)$ includes all subsets $\mathcal{DAS}_x(c_k)$, where $x \in \mathcal{D}iff(c_k)$.

Continuing with our running example, suppose we aim to identify all Decisive Attribute Sets that allow f to predict the class $c_k = \{C_2\}$. The first step is to identify the formal concepts whose intent includes the class c_2 in $\mathcal{L}'(K')$:

$\mathcal{B}(\{c_2\}) = \{fc_2\}$. $|\mathcal{B}(c_2)| = 1$, this implies that only one formal concept includes c_2 in its intent.
In this case, the Decisive Attribute Set for predicting c_2 is simply the condition of this unique formal concept, namely:
$\mathcal{DAS}(c_2) = \{a_3\}$
Similarly, for the class c_3, it appears in a single formal concept, $\mathcal{B}(\{c_3\}) = \{fc_4\}$, and thus:
The Decisive Attribute Set for predicting c_3 is:
$\mathcal{DAS}(c_3) = \{a_1, a_2\}$
For the class $c_k = c_1$, it appears in two formal concepts, which are fc_3 and fc_4. Let us compute $\mathcal{D}iff(c_1)$:
$\mathcal{D}iff(c_1) = \bigcup \mathcal{E}xt(fc_i) \backslash \bigcap \mathcal{E}xt(fc_i), \quad \forall fc_i \in \mathcal{B}(\{c_1\})$,
resulting in $\mathcal{D}iff(c_1) = \{x_2, x_3\}$.

For each $x \in \mathcal{D}iff(c_1)$, compute $\mathcal{DAS}_x(c_1)$.
For the instance x_2, $\mathcal{DAS}_{x_2}(c_1) = \{a_1\}$
For the instance x_3, $\mathcal{DAS}_{x_3}(c_1) = \{a_1\}$
In this case, $\mathcal{DAS}_{x_2}(c_1) = \mathcal{DAS}_{x_3}(c_1)$.
Thus, the complete set of Decisive Attribute Sets for predicting $y = \{c_1\}$ is:
$\mathcal{DAS}(c_1) = \mathcal{DAS}_{x_2,x_3}(c_1) = \{a_1\}$

6.2 Significant attribute set

Attributes that are present in at least one Decisive Attribute Set are considered crucial, as they play a key role in determining the classification of an instance into a specific class. Each $\mathcal{DAS}$ corresponds to the minimal set of attributes that is required for accurate classification, providing a clear justification for the classification of a data instance.

The Significant Attribute Set for a class c_k (denoted as $\mathcal{SSA}(c_k)$) comprises all attributes that appear in at least one Decisive Attribute Set for that class, which effectively forms the union of these sets. For a set of classes y, the Significant Attribute set is defined as the union of the significant attribute sets of each class in y:

$$\mathcal{SSA}(y) = \bigcup_{c_k \in y} \mathcal{SSA}(c_k).$$

The Significant Attribute Set for $y = \{c_1\}$ is the union of all Decisive Attribute Sets for the class c_1. Thus, $\mathcal{SSA}(c_1) = \{a_1\}$. In the same way, the Significant Attribute Set for $y = c_3$ is the union of all Decisive Attribute Sets for the class c_3. Thus,
$\mathcal{SSA}(c_3) = \{a_1, a_2\}$.

For both classes c_1 and c_3, the combined Significant Attribute Set is:
$\mathcal{SSA}(c_1, c_3) = \mathcal{SSA}(c_1) \cup \mathcal{SSA}(c_3)$:
$\mathcal{SSA}(c_1, c_3) = \{a_1, a_2\}$.

6.3 Symbolic explanations and feature importance in multi-label predictions

Recall that our approach for explaining multi-label classification is model-agnostic and provides both *symbolic* explanations and feature importances.

In this section, we illustrate, through a real example from the medical field, how the previously defined $\mathcal{DAS}$ (Decisive Attribute Sets) and $\mathcal{SSA}$ (Significant Attribute Set) can be computed and interpreted in practice. This example deals with diagnosing some respiratory diseases (labels) given the symptoms observed in patients (attributes). For the sake of simplicity, we consider only four diseases: Asthma (d_1), Bronchiolitis (d_2), COPD (d_3), and COVID (d_4).

These diseases generally manifest through the following symptoms: Dry cough (s_1), Loose cough (s_2), Shortness of breath (s_3), Wheezing (s_4), Fever (s_5), Headaches (s_6), Loss of taste (s_7), Curvatures (s_8), and Dyspnoea (s_9).

Table 3: Multi-label predictions generated by the black-box model f for the neighborhood of the target instance x under explanation.

	Symptoms									Diseases			
$\mathcal{O}$	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	d_1	d_2	d_3	d_4
1	1	0	1	1	1	0	0	0	0	1	1	0	0
2	0	1	0	1	1	0	0	1	1	1	0	1	0
3	1	0	1	0	1	1	1	1	0	1	0	0	1
4	1	0	0	0	1	1	1	1	0	0	0	0	1
5	0	1	1	0	0	0	0	1	1	0	0	1	0
6	1	0	1	0	1	1	1	1	0	1	1	0	1
7	0	0	0	0	1	1	1	1	0	0	0	0	1
8	1	0	1	1	1	1	0	1	0	1	1	0	1
9	1	0	1	0	0	0	0	1	0	0	1	0	0
10	1	0	1	0	1	1	1	1	0	0	1	0	1
11	0	1	1	1	1	0	0	0	1	1	1	1	0
12	0	1	0	0	1	0	0	0	1	0	0	1	0
13	1	0	1	0	1	1	1	1	0	0	0	0	1
14	1	0	1	1	0	0	0	0	0	1	0	0	0

After transforming these multi-label data into a formal context, we obtain the following reduced set of formal concepts $\mathcal{L}'(K')$:

fc_1: ⟨{11}, {Asthma, Bronchiolitis, COPD}, {fever, Dyspnoa, Shortness of breath, Wheezing, Loose cough}⟩
fc_2: ⟨{2, 11}, {Asthma, COPD}, {Fever, Loose cough, Dyspnoa, Wheezing}⟩
fc_3: ⟨{8, 6}, {Asthma, Bronchiolitis, Covid}, {Fever, Headache, Shortness of breath, Dry cough, Curvature}⟩
fc_4: ⟨{8, 3, 6}, {Asthma, Covid}, {Fever, Headache, Shortness of breath, Dry cough, Curvature}⟩
fc_5: ⟨{8, 10, 6}, {Bronchiolitis, Covid}, {fever, Headache, Shortness of breath, Dry cough, Curvature}⟩
fc_6: ⟨{8, 1, 11}, {Asthma, Bronchiolitis}, {Fever, Shortness of breath}⟩
fc_7: ⟨{2, 11, 12, 5}, {COPD}, {Loose cough, Dyspnoa}⟩
fc_8: ⟨{1, 2, 8, 11, 14}, {Asthma}, {wheezing}⟩
fc_9: ⟨{1, 2, 3, 6, 8, 11}, {Asthma}, {Fever}⟩
fc_{10}: ⟨{1, 3, 6, 8, 11, 14}, {Asthma}, {Shortness of breath}⟩
fc_{11}: ⟨{1, 6, 8, 9, 10, 11}, {Bronchiolitis}, {Shortness of breath}⟩
fc_{12}: ⟨{3, 4, 6, 7, 8, 10, 13}, {Covid}, {Fever, Headache, Curvature}⟩

Given a data instance x and the prediction $f(x)$, we are first interested in symbolic explanations that are *sufficient reasons*. Broadly speaking, they refer to the subset of features in x that are sufficient to predict $y = f(x)$.

Decisive Attribute set ($\mathcal{DAS}$): Assume we want to compute the set of all Decisive attribute Sets that enable f to predict the class $c_k = \{\text{COVID}\}$ (also denoted as d_4). First, we select the formal concepts having in their intent the class:

$$\mathcal{B}(\{\text{COVID}\}) = \{fc_3, fc_4, fc_5, fc_{12}\}$$

Next, compute the difference between extents:

$$\mathcal{D}iff(\text{COVID}) = \bigcup \mathcal{E}xt(fc_i) \setminus \bigcap \mathcal{E}xt(fc_i) = \{3, 4, 7, 10, 13\}$$

For each instance $x \in \mathcal{D}iff(\text{COVID})$, we compute its decisive attributes $\mathcal{DAS}_x(\text{COVID})$:

- For instances $\{10, 3\}$: $\mathcal{DAS}_{10,3}(\text{COVID}) = \{\text{Fever, Headache, Shortness of breath, Dry cough, Curvature}\}$
- For instances $\{4, 13, 7\}$: $\mathcal{DAS}_{4,13,7}(\text{COVID}) = \{\text{Fever, Headache, Curvature}\}$

The complete set of decisive attributes for the class COVID is:

$$\mathcal{DAS}(\text{COVID}) = \{\mathcal{DAS}_{4,13,7}(\text{COVID}), \mathcal{DAS}_{10,3}(\text{COVID})\}$$

These sets represent the minimal symptom subsets sufficient for model f to predict the class $y = \{\text{COVID}\}$ for the corresponding instances.

Significant Attribute Set:
The Significant Attribute Set for $y = \{\text{COVID}\}$ is defined as the union of all Decisive Attribute Sets for that class:

$$\mathcal{S}sa(\text{COVID}) = \{\text{Fever, Headache, Curvature, Dry cough, Shortness of breath}\}$$

This set contains all symptoms appearing in at least one Decisive Attribute Set associated with predicting the class *COVID*. In other words, each of these symptoms played a role in at least one instance where the model f predicted *COVID*. The set $\mathcal{SSA}(\text{COVID})$ therefore provides a global, symbolic explanation of the disease: it captures the complete pool of symptoms that have been sufficient, in different combinations, to trigger the model's decision for *COVID*.

6.3.1 Introducing attribute importance

While symbolic explanations such as $\mathcal{DAS}$ and $\mathcal{SSA}$ provide insight into which attributes are sufficient for predicting specific labels, they do not directly quantify the relative importance of each attribute in the overall decision-making process.

In practical applications, particularly in sensitive domains such as healthcare, it is crucial not only to identify the features that justify a prediction, but also to assess how strongly each feature contributes to it. This motivates the need for a *measure of feature importance*, which can help users grasp the influence of each individual attribute in the model's decisions.

In the following, we build upon the extracted formal concepts to propose an interpretable way to quantify attribute importance in the multi-label setting.

To stay within our symbolic and concept-based explanation framework, we propose a simple and transparent method to estimate how important each attribute is in predicting a given class. This idea is based on the formal concepts already extracted from the data.

The key intuition is simple: an attribute is considered important if it appears often in the different sets of features

that are sufficient to justify the prediction of a class. The more frequently an attribute appears in such explanations, the greater its influence on the model's decision for that class.

Next, we define this measure in a precise way and explain how it can be computed and interpreted using the formal concepts.

Attribute Importance Score.

To address this need, we propose a simple and interpretable measure of feature importance that is consistent with the concept-based explanations introduced earlier. Rather than relying on simple frequency counts or cumulative support, our measure aggregates the set of unique instances covered by all formal concepts that include both the attribute and the class. This ensures that each instance is counted only once, even if it appears in multiple overlapping concepts.

Let F be the set of formal concepts, a an attribute, and c a class label. We define:

- $\mathcal{F}_{a,c}$: the subset of concepts in F where both attribute a and class c appear (in the condition and the intent respectively),
- $\mathcal{F}_c$: the subset of concepts in F where class c appears in the intent (regardless of attribute a),
- $\text{ext}(\mathcal{F})$: the union of the extents of all concepts in $\mathcal{F}$, i.e., all distinct instances involved.

The importance score of attribute a for class c is then defined as:

$$\text{Imp}(a, c) = \frac{|\text{ext}(\mathcal{F}_{a,c})|}{|\text{ext}(\mathcal{F}_c)|}$$

This ratio reflects how widely attribute a contributes to justifying the class c across all its supporting concepts. A higher score indicates that a is involved in explaining a larger proportion of the instances predicted to belong to class c.

Illustrative example.

We illustrate the computation of the importance score for attributes with respect to the class *COPD*, using the formal concepts introduced earlier.

Let $c = \text{COPD}$, and let $\mathcal{F}$ denote the set of formal concepts extracted from the dataset. A formal concept is defined as a triple ⟨extent, intent, condition⟩, where the extent is the set of instances, the intent is the set of predicted class labels, and the condition is the set of attributes shared by these instances. For instance, the following concepts include the class COPD in their intent: Thus, we obtain:

$$\text{ext}(F_c) = \{2, 5, 11, 12\} \quad \text{and} \quad |\text{ext}(F_c)| = 4$$

Now, we compute the importance for several attributes:

- **Loose cough:** Appears in all three concepts $\Rightarrow$ $\text{ext}(F_{\text{Loose cough},c}) = \{2, 5, 11, 12\} \Rightarrow \text{Imp} = \frac{4}{4} = 1.0$
- **Dyspnoea:** Appears in all three concepts $\Rightarrow$ $\text{ext}(F_{\text{Dyspnoea},c}) = \{2, 5, 11, 12\} \Rightarrow \text{Imp} = \frac{4}{4} = 1.0$
- **Wheezing:** Appears in fc_1 and fc_2: $\text{ext}(F_{\text{Wheezing},c}) = \{2, 11\} \Rightarrow \text{Imp} = \frac{2}{4} = 0.5$
- **Fever:** Appears in fc_1 and fc_2: $\text{ext}(F_{\text{Fever},c}) = \{2, 11\} \Rightarrow \text{Imp} = \frac{2}{4} = 0.5$
- **Shortness of breath:** Appears only in fc_1: $\text{ext}(F_{\text{Shortness of breath},c}) = \{11\} \Rightarrow \text{Imp} = \frac{1}{4} = 0.25$

Features such as *Loose cough* and *Dyspnoea* exhibit the highest importance scores, since they appear in all formal concepts related to the class *COPD*. Consequently, whenever these attributes are present in the explanations, they are the most decisive in the prediction of *COPD*. In contrast, attributes like *Fever* and *Shortness of Breath* occur only in a subset of these concepts, while the other attributes do not contribute at all to the prediction of this class.

Feature importance visualization

To provide a clearer understanding of the influence of each feature on the model's predictions, we visualize the importance scores computed for all features and each class label.

The following diagrams display the relative importance of features with respect to the predicted classes: **Asthma**, **Bronchiolitis**, **COPD**, and **COVID**. These visualizations are based on the formal concepts extracted during the explanation process, and they reflect how strongly each feature contributes to each individual class.

A higher bar indicates that the corresponding feature appears more frequently and significantly in the concepts associated with that class. This graphical representation supports the interpretability of the model by highlighting the most influential symptoms for each diagnosis.

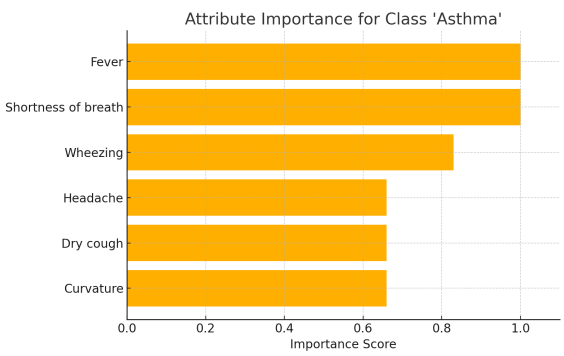

Figure 6: Feature importance for the class *Asthma*

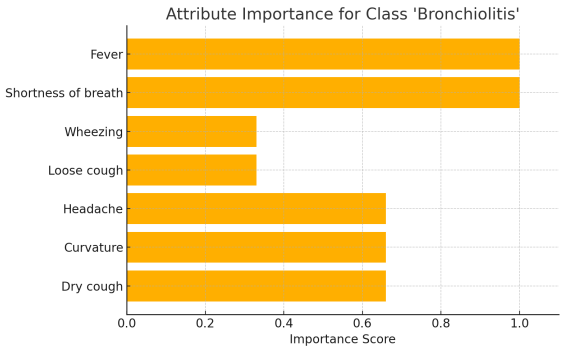

Figure 7: Feature importance for the class *Bronchiolitis*

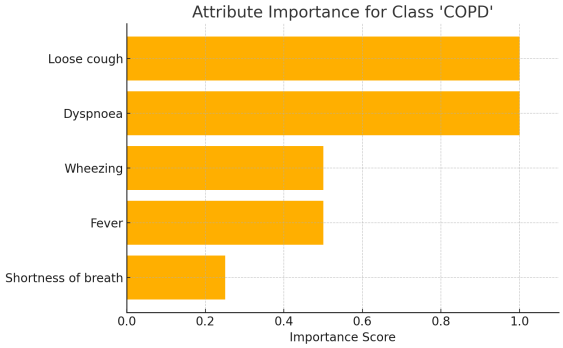

Figure 8: Feature importance for the class *COPD*

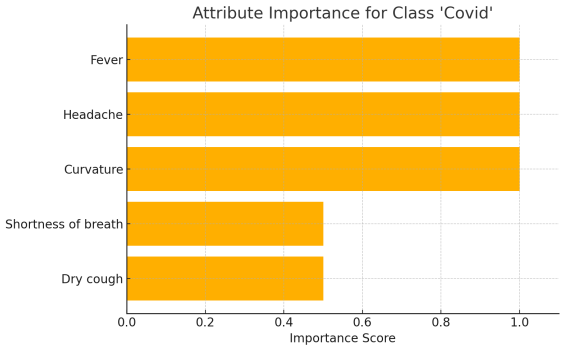

Figure 9: Feature importance for the class *COVID*

The proposed importance score offers a simple yet effective way to quantify how relevant each feature is to a specific class prediction. Unlike purely symbolic explanations, which identify sets of sufficient features, this measure provides a *numerical assessment* that reflects both the *frequency* and *specificity* of each feature in relation to the class.

Beyond its technical value, this measure is highly beneficial for the end users. It provides an interpretable, class-wise ranking of features, helping decision-makers, such as healthcare professionals or other domain experts, to better understand which features significantly influence the model's predictions. This contributes to building trust, enhancing transparency, and supporting informed decision-making.

6.4 Extending explanations

hus far, we have primarily focused on $\mathcal{DAS}$ explanations, which provide sufficient reasons for predicting a given label. Building on the concept of $\mathcal{DAS}$, we have also defined $\mathcal{SSA}$ explanations (which correspond to relevant features for a given prediction) derived from $\mathcal{DAS}$ explanations. Using these as a foundation, other explanations can be derived to address different types of XAI queries. Notably, we highlight the following:

– *Irrelevant attribute set $\mathcal{IAS}$ explanation :* An irrelevant feature is one that does not appear in any explanation of a prediction. Such features have no impact on the prediction y. It is clear that $\mathcal{IAS}$ can be easily determined as the features that are not included in $\mathcal{SSA}$:

$$\mathcal{IAS}(c_k) = \mathcal{P} \setminus \mathcal{SSA}(c_k)$$

where $\mathcal{P}$ denotes the set of all attributes.

– *Necessary attribute set $\mathcal{NAS}$ explanation :* Intuitively, a necessary feature is one that appears in every $\mathcal{DAS}$ explanation, making it essential for obtaining the prediction. This could indicate a highly relevant feature or a bias in the data, as the prediction relies on that feature. To extract the $\mathcal{NAS}$, it is sufficient to consider the intersection of all the $\mathcal{DAS}$ to directly identify them.

$$\mathcal{NAS}(c_k) = \cap\mathcal{DAS}(c_k)$$

– *$\mathcal{DAS}$-rules* : One might be interested in other types of explanations derived from $\mathcal{DAS}$. A challenge that arises when considering all possible explanations is how to filter and select them based on specific criteria of interest or quality. To address this, we propose the creation of $\mathcal{DAS}$-rules, which are sufficient reasons similar to the $\mathcal{DAS}$ explanations, but can also offer valuable insights into the *strength* or *prevalence* of the rule.

$\mathcal{DAS}$-rules are similar to anchor rules [41] and follow the form: IF $\mathcal{DAS}_i$(class) THEN predict class while indicating the percentage of data instances in the chosen neighborhood that comply with this rule. These IF-THEN rules capture local conditions for predictions. A $\mathcal{DAS}$-rule defines the prediction locally, meaning that changes to other features may not alter the prediction. While $\mathcal{DAS}_i(class)$ applies to one instance, a $\mathcal{DAS}$-rule covers its neighborhood.
Formally, for a given class c_i, a $\mathcal{DAS}$-rule is defined as an IF-THEN rule:

$$\text{IF } \mathcal{DAS}(c_i) \text{ THEN predict } c_i$$

where $\mathcal{DAS}(c_i) \in \mathcal{DAS}$ represents one of the Decisive Attribute Sets enabling the classifier f to predict class c_i. The number of $\mathcal{DAS}$-rules corresponds to the number

of distinct $\mathcal{DAS}$ explanations for a specific class c_i. The coverage metric evaluates how frequently each $\mathcal{DAS}$-rule appears within the neighborhood of the instance being explained. A $\mathcal{DAS}$-rule is deemed relevant if the number of instances sharing the rule and yielding the same prediction as the instance surpasses a defined threshold. This measure, termed $\mathcal{DAS}$-rule coverage, reflects the percentage of instances in the neighborhood that agree with a specific explanation. To extract $\mathcal{DAS}$-rules, two steps are involved: (1) enumerate $\mathcal{DAS}$ explanations for the instance x to be explained, and (2) assess each $\mathcal{DAS}$ explanation to determine if its coverage (i.e., the percentage of agreeing instances in x's neighborhood) exceeds a predefined threshold t. The number of $\mathcal{DAS}$-rules cannot exceed the number of $\mathcal{DAS}$ explanations, and extracting these rules only cannot exceed the number ofrequires scanning x's neighborhood, initially generated during the computation of $\mathcal{DAS}$ explanations. A $\mathcal{DAS}$-rule is valid at threshold t if:

$$\mathcal{DAS}(c_i) \Rightarrow c_i$$

with Coverage $(\mathcal{DAS}(c_i) \Rightarrow c_i) \geq t$.

7 Experimental study

To evaluate the practical feasibility of our method, we performed experiments on three widely-used multi-label datasets. The first set of experiments examines the relatively small Stack Overflow dataset, which serves as a case study to demonstrate the types of explanations our approach can generate. The remaining two experiments, conducted on the Yelp and TMC2007-500 dataset, aim to analyze the impact of some parameters on the number of explanations, their size, and the time needed for enumeration.

7.1 Experimental setup

In order to present our experimental evaluation, we first recall the main steps of our explanation method and then present the experimental setup.

1. *Generate local contexts*: In this step, we generate the formal contexts around the instance to be explained. This task is performed in our experiments using the k-Nearest Neighbors (KNN) algorithm. This step is detailed in Section 5.1.
2. *Identify and extract all formal concepts*;
3. *Minimize formal concepts* (see Section 5.3 for more details on these last two steps);
4. *Compute the explanations* (namely, the $\mathcal{DAS}$ and $\mathcal{SSA}$).

7.1.1 Datasets

To evaluate the effectiveness of our explanation method, we employed three multi-label datasets. The first is a relatively small dataset derived from the Stack Overflow platform, designed to clearly illustrate the behavior of our approach. Its reduced size and controlled structure enable fine-grained analysis and visualization of the generated explanations. The other two datasets, Yelp and TMC2007-500 are widely used benchmarks in the multi-label learning literature and serve to assess the scalability and robustness of our method under more realistic and complex conditions.

Stack overflow dataset This dataset was specifically constructed for our case study on explainable multi-label classification. It contains 500 programming-related questions collected from the Stack Overflow platform. Each instance is represented by a 100-dimensional binary vector, where each feature encodes the presence or absence of selected terms from the question's title and description. Standard text preprocessing techniques were applied, including tokenization, stop-word removal, and vocabulary filtering.

The label space consists of five widely used technologies: *Asp.net*, *Java*, *MySQL*, *PHP*, and *Python*. Each question may be associated with one or more of these labels, forming a typical multi-label setting. We deliberately constructed this small dataset to facilitate the visualization and analysis of the explanations produced by our method—something that would be considerably more difficult to achieve with large-scale datasets.

The dataset is publicly available at: `https://github.com/hakimradja/fca-explainability/tree/main/stack-overflow`

Yelp dataset The Yelp dataset was obtained from the COMETA repository (`https://cometa.ujaen.es/datasets/Yelp`) and originates from the Yelp Dataset Challenge [47]. It comprises 10,806 instances, each represented by 676 binary features, including 5 target labels and 671 descriptive attributes. The input features are constructed from a bag-of-words representation of textual reviews, where each attribute encodes the presence or absence of a specific token following standard text preprocessing.

The classification task is inherently multi-label: each review may belong to multiple predefined categories (e.g., food, service, ambiance). This dataset presents a high-dimensional, sparse input structure, which makes it a suitable benchmark for evaluating the performance and interpretability of multi-label classification models.

TMC2007-500 dataset The TMC2007-500 dataset was also obtained from the COMETA repository (`https://cometa.ujaen.es/datasets/tmc2007_500`). It is derived from the Aviation Safety Reporting System (ASRS) and contains 2,200 textual incident reports related to aviation safety. Each report is encoded as a binary vector of 500 features, extracted using a bag-of-words representation following standard preprocessing steps (tokenization, stop-word removal, and term selection).

The label space includes 22 distinct safety categories, and each report may be annotated with multiple labels reflecting the multifactorial nature of aviation incidents (e.g., human factors, procedural deviations, environmental hazards). The dataset exhibits an with an average label cardinality of 2.22 (i.e., the average number of labels per instance), indicating that each report is typically associated with more than two categories. This dataset serves as a realistic benchmark for evaluating the scalability and explanatory power of multi-label classification methods [48].

7.1.2 Parameter settings

In all experiments, we used a consistent 70/30 split between training and testing data. We considered three widely used multi-label classification algorithms known for their effectiveness: Label Powerset (LP), Random k-Labelsets (RAkEL), and Multi-Label k-Nearest Neighbors (ML-KNN). Each of these methods relies on specific parameters, which we set following commonly accepted practices in the literature.

For the Label Powerset (LP) method, we used a Random Forest as the base classifier, configured with 100 trees (`n_estimators = 100`). Other hyperparameters of the Random Forest remained at their default values. This combination provides a balance between predictive performance and robustness.

The RAkEL ensemble method was configured with standard parameter choices: the number of LP models was set to 10 (`model_count = 10`), and each LP model was trained on random subsets of 3 labels (`labelset_size = 3`). We used a Decision Tree as the base classifier for each individual model. These values are widely used in benchmark studies and provide a trade-off between diversity and computational efficiency.

For the ML-KNN method, we used the default settings from the `scikit-multilearn` library. The number of neighbors was set to $k = 10$, and Laplace smoothing was applied with a parameter $s = 1.0$. All neighbors were weighted equally in distance calculations (`weighting = 'uniform'`).

7.1.3 Neighborhood selection

To generate local contexts around the instance to be explained, we applied the K-Nearest Neighbors (KNN) algorithm using the Euclidean distance metric. The size of the neighborhood, denoted by k, directly affects the trade-off between the locality of the explanations and the generality of the extracted concepts.

More concretely:

- Smaller neighborhoods (e.g., $k = 10$) lead to more specific concepts that are closely tailored to the instance, but less generalizable to other cases.
- Larger neighborhoods (e.g., $k = 100$) yield more general concepts based on a wider range of instances, but those concepts may be less precise for the instance being explained.

We empirically explored several values of k, ranging from 10 to 100. We selected the values $\{20, 50, 100\}$ to reflect three representative levels of locality in explanations:

- $k = 20$: small neighborhoods yielding more specific, fine-grained concepts,
- $k = 50$: moderate size allowing a balance between specificity and generality,
- $k = 100$: larger neighborhoods resulting in more general and potentially more stable concepts.

Rather than exploring every possible value of k between 10 and 100, we adopted a non-uniform sampling strategy to reduce redundancy and keep the explanation process computationally tractable. Preliminary tests showed that intermediate values (e.g., 30, 40, 60, 70, 80...) did not yield substantially different conceptual structures or explanation patterns. Thus, we chose a compact yet representative set of neighborhood sizes.

7.1.4 Concept extraction and explanation generation

After generating the formal context representing the neighborhood of the instance to be explained, we apply the Next Closure algorithm to extract the complete set of formal concepts. Since the number of concepts can be large and some may be redundant, we filter the results to retain only the most informative concepts. From this reduced set, we derive the explanations in terms of *Decisive Attribute Sets (DAS)* and *Significant Attribute Sets (SSA)*.

To summarize the previously described explanation process, the following algorithms outline the main steps involved in generating symbolic explanations from a target instance using Formal Concept Analysis (FCA).

	Title	Tags
0	Flask-SQLAlchemy - When are the tables/databas...	['python', 'mysql']
1	Combining two PHP variables for MySQL query	['php', 'mysql']
2	Counting' the number of records that match a c...	['php', 'mysql']
3	Insert new row in a table and auto id number. ...	['php', 'mysql']
4	Create Multiple MySQL tables using PHP	['php', 'mysql']
...	...	...

Figure 10: Example question from the Stack Overflow dataset

Algorithm 3 FCA-based Explanation Generation — Data Preparation

Input
- Labeled dataset (X, Y) with binary attributes and multi-label targets;
- Multi-label classifier (e.g., Label Powerset with Random Forest);
- Target instance x to be explained;
- Neighborhood size k (e.g., $k = 20$)

Output
- Binary formal context for x's neighborhood

1. Split (X, Y) into training and test sets.
2. Train the classifier on the training data.
3. Select a target instance x from the test set.
4. Find the k nearest neighbors of x and predict their labels.
5. Concatenate neighbors and their predicted labels into a local dataset.
6. Transform the local dataset into a binary formal context:
   (a) Create compound attributes (c_j, a_i).
   (b) Encode the binary presence of each compound attribute.

Algorithm 4 FCA-based Explanation Generation (Part 2: Concept Extraction & Explanation)

Input
- Binary formal context from Part 1

Output
- Reduced set of formal concepts;
- For each label c_k, we derive its Decisive Attribute Sets (DAS) as well as its Significant Attribute Set (SSA(c_k));
- For each label c_k, we compute the importance degree of each attribute involved in its prediction.

1. Generate all formal concepts using the NextClosure algorithm.
2. Reduce the set by removing redundant concepts and those with low explanatory value.
3. For each label, compute the Decisive Attribute Set ($\mathcal{DAS}$) and the Significant Attribute Set ($\mathcal{SSA}(c_k)$).
4. Return the reduced set of concepts, along with $\mathcal{DAS}$ and $\mathcal{SSA}(c_k)$ for each label.
5. Return the attribute importance values for each class.

7.2 Case study on stack overflow data

This section presents a case study on the Stack Overflow dataset, derived from a widely used platform of programming questions and answers. Each question includes a *Title*, *Description*, and *Tags*, as shown in Figure 11.

Our prediction task involves assigning appropriate tags (labels) to a question based on its title and description. This resulted in a multi-label dataset characterized by:

- 499 instances,
- 100 features,
- 5 labels (*Asp.net, Java, MYsql, Php, Python*).

As previously introduced, we focus on three prominent multi-label classification methods: *Label Powerset (LP)*, *ML-KNN*, and *RAKEL*, which respectively represent problem transformation, problem adaptation, and a hybrid ensemble approach. To explain the predictions made by these classifiers for individual instances, we calculated both the **Decisive Attribute Sets** ($\mathcal{DAS}$) and the **Significant Attribute Set** ($\mathcal{SSA}$) for each class. The decisive attributes reveal the specific features that were essential in the decision of each classifier, while the significant attributes highlight the most influential features across predictions for that class. Our experimental approach involved selecting an instance at random and examining its classification outcomes through progressively larger neighborhoods, adjusting the neighborhood size K from 10 to 100 in increments of 10. This produced 10 progressively larger local formal contexts, each consisting of K nearest neighbors. For each classifier, we analyzed these local contexts, observing how predictive patterns evolved as the neighborhood size changed.

Illustrative example: class mysql.1 with LP classifier

To analyze this process, we present a detailed example of predictions generated by the LP classifier for an instance within the **mysql.1** class. Here, we analyze a local formal context containing the 50 nearest neighbors (i.e., $K = 50$). We provide explanations in terms of both the $\mathcal{DAS}$ and $\mathcal{SSA}$, identifying the primary attributes driving the classifier's decisions. The dataset comprises five classes. For each, we specify the distinct Decisive Attribute Sets ($\mathcal{DAS}$) leveraged by the classifier, alongside the number of significant attributes. This analysis provides insight into the attribute combinations that have the greatest impact on the predictions for each class.

Decisive attribute sets for mysql.1 class

For the **mysql.1** class, we observed the following Decisive Attribute Sets:

- $\mathcal{DAS}(mysql.1)$={{'creating', 'site', 'aspnet', 'maps'}, {'mysql'}, {'sqlalchemy'}, {'database', 'python'}, {'table'}, {'pymysql'}, {'database'}, {'database', 'python', 'mysql'}, {'position', 'can'}, {'python', 'mysql'}, {'php'}, {'change'}, {'database', 'mysql', 'sqlalchemy'}, {'login'}, {'creating'}, {'string', 'arguments'}, {'database', 'storing', 'variable'}}

The **Significant Attribute Set** for mysql.1, representing attributes frequently critical across predictions, is:

– $\mathcal{SSA}(mysql.1)$={'python', 'sqlalchemy', 'storing', 'php', 'login', 'site', 'maps', 'database', 'creating', 'table', 'arguments', 'string', 'change', 'mysql', 'position', 'can', 'aspnet', 'variable', 'pymysql'}

Among the 50 nearest neighbors, the LP classifier assigned 36 instances to the mysql.1 class, using 17 unique $\mathcal{DAS}$ in this process. The distribution of instances across these sets is outlined in Table 4.

Table 4: Number of instances per $\mathcal{DAS}$ (class: mysql.1, classifier: LP, neighbors $K = 50$)

Instance count	$\mathcal{DAS}$ used
9	{'sqlalchemy'}
3	{'mysql'}
3	{'pymysql'}
3	{'database'}
3	{'database', 'python', 'mysql'}
2	{'creating', 'site', 'aspnet', 'maps'}
2	{'table'}
2	{'php'}
1	{'database', 'python'}
1	{'python', 'mysql'}
1	{'change'}
1	{'database', 'mysql', 'sqlalchemy'}
1	{'login'}
1	{'creating'}
1	{'string', 'arguments'}
1	{'database', 'storing', 'variable'}
1	{'position', 'can'}

To complement the symbolic explanations, we quantified the relative importance of the features contributing to the class `mysql.1` by leveraging a concept-based metric. For each attribute, its importance score was computed as the proportion of instances assigned to the `mysql.1` class that appear in at least one formal concept whose intent includes both the class label and the attribute. This metric thus captures how broadly an attribute participates in the justification of the classifier's decisions, reflecting both its recurrence across instances and its discriminative relevance.
As shown in Figure 11, the attributes *sqlalchemy*, *database*, and *mysql* obtained the highest importance scores. Their strong presence indicates that they are not only sufficient in isolated cases but also recurrently involved across multiple instances, making them decisive markers for the prediction of the class `mysql.1`. In contrast, other attributes such as *python*, *creating*, and *pymysql* exhibit moderate contributions, being involved only in a subset of explanations. Less frequent terms, including *php*, *table*, or *login*, contribute minimally to the explanation and can be considered secondary signals.
This numerical layer of explanation complements symbolic reasoning by providing a ranked perspective on the most influential features. Such a ranking not only highlights the decisive attributes but also offers end users—such as developers or database administrators—a clearer understanding of which features most strongly influence the model's predictions, thereby enhancing interpretability and transparency.

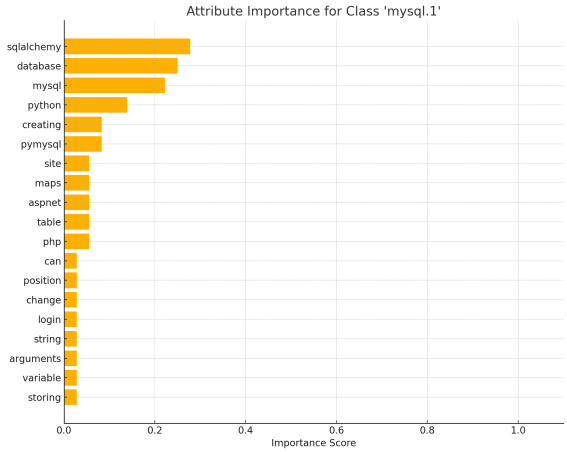

Figure 11: Feature importance for the class `mysql.1`.

$\mathcal{DAS}$**-rules**: We selected an instance randomly and generated a neighborhood of 50. The classifier used was LP. The $\mathcal{DAS}$-rules for the class mysql.1 are obtained from the $\mathcal{DAS}$ used to predict this class presented in Table 4.
Among the 50 nearest instances to the one being explained, 36 were predicted by the LP classifier in the class mysql.1. This was done using 17 different Decisive Attribute Sets ($\mathcal{DAS}$). Table 5 shows the $\mathcal{DAS}$-rules obtained with the coverage (proportion) of each rule.

Table 5: $\mathcal{DAS}$-rules obtained for the mysql.1 class of the Stack Overflow dataset (number of neighbors $K = 50$).

Coverage	$\mathcal{DAS}$-rule
0.18	$'sqlalchemy' \Rightarrow mysql.1$
0.06	$'mysql' \Rightarrow mysql.1$
0.06	$'pymysql' \Rightarrow mysql.1$
0.06	$'database' \Rightarrow mysql.1$
0.06	$'database' \wedge' python' \wedge' mysql' \Rightarrow mysql.1$
0.04	$'creating' \wedge' site' \wedge' aspnet' \wedge' maps' \Rightarrow mysql.1$
0.04	$'table' \Rightarrow mysql.1$
0.04	$'php' \Rightarrow mysql.1$
0.02	$'database' \wedge' python' \Rightarrow mysql.1$
0.02	$'python' \wedge' mysql' \Rightarrow mysql.1$
0.02	$'change' \Rightarrow mysql.1$
0.02	$'database' \wedge' mysql' \wedge' sqlalchemy' \Rightarrow mysql.1$
0.02	$'login' \Rightarrow mysql.1$
0.02	$'creating' \Rightarrow mysql.1$
0.02	$'string' \wedge' arguments' \Rightarrow mysql.1$
0.02	$'database' \wedge' storing' \wedge' variable' \Rightarrow mysql.1$
0.02	$'position' \wedge' can' \Rightarrow mysql.1$

Table 6: Number of Decisive Attribute Sets ($\mathcal{DAS}$) per class for a selected instance, based on varying neighborhood sizes from 10 to 100. Experiments conducted using LP, ML-KNN, and RAKEL classifiers

Neighbors	Classifier	Asp.net.1	Java.1	Mysql.1	Php.1	Python.1
10	LP	1	0	1	0	0
	ML-KNN	1	1	3	0	2
	RAKEL	1	0	1	0	0
20	LP	1	0	4	0	3
	ML-KNN	1	1	4	0	3
	RAKEL	1	0	4	0	3
30	LP	1	1	8	1	7
	ML-KNN	1	1	9	1	6
	RAKEL	1	1	8	1	7
40	LP	1	1	12	3	9
	ML-KNN	1	1	14	5	8
	RAKEL	1	1	12	3	10
50	LP	1	1	17	6	11
	ML-KNN	1	1	18	8	9
	RAKEL	1	1	17	6	12
60	LP	1	1	23	10	11
	ML-KNN	1	1	21	10	10
	RAKEL	1	1	23	10	17
70	LP	1	1	31	12	20
	ML-KNN	1	1	30	12	16
	RAKEL	1	1	31	12	24
80	LP	1	1	35	12	24
	ML-KNN	1	1	33	12	17
	RAKEL	1	1	33	12	28
90	LP	1	1	37	12	25
	ML-KNN	1	1	34	14	17
	RAKEL	1	1	37	12	30
100	LP	1	1	45	15	28
	ML-KNN	1	1	44	19	21
	RAKEL	1	1	45	15	34

For example, the $\mathcal{DAS}$-rule The $\mathcal{DAS}$-rule "IF 'sqlalchemy' THEN predict `mysql.1`" with a coverage of 0.18 means that 18% of the 50 data instances close to the one being explained (9 out of the 50 instances) were predicted by the multi-label LP classifier in the `mysql.1` class when the 'sqlalchemy' attribute is present.
To trace the behavior of each classifier, Table 6 and Table 7 below summarize the results obtained when we applied three different algorithms — Label Powerset (LP), ML-KNN, and RAKEL on our dataset. For each algorithm, we provide both the number of Decisive Attribute Sets ($\mathcal{DAS}$) used and the percentage of Significant Attributes ($\mathcal{SSA}$) across each class in the dataset.

Analyzing the data from Table 6 reveals that the number of Decisive Attribute Sets ($\mathcal{DAS}$) for each class varies depending on the classifier. This indicates that each algorithm employs a unique selection of attributes to make predictions for a given class or set of classes, indicating that each classifier's decision-making relies on different aspects of the data. For instance, when using a neighborhood of 10 instances, both the LP and RAKEL classifiers converged on the same Decisive Attribute Set for predicting the $mysql.1$ class:

- $\mathcal{DAS}(mysql.1)$ = {'creating', 'maps', 'aspnet', 'site'}

In contrast, the ML-KNN classifier used three distinct Decisive Attribute Sets for the same class:

- $\mathcal{DAS}_1(mysql.1) = \{\text{creating, maps, aspnet, site}\}$
- $\mathcal{DAS}_2(mysql.1) = \{\text{sqlalchemy}\}$
- $\mathcal{DAS}_3(mysql.1) = \{\text{database, web, python, server, connection}\}$

This variability suggests that ML-KNN considers multiple unique combinations of attributes, reflecting a broader set of criteria in its predictions for the$mysql.1$ class. Furthermore, we observe that the number of $\mathcal{DAS}$ per class changes as the neighborhood size increases. This occurs in two distinct ways:

1. Constant $\mathcal{DAS}$ Count: For some classes, such as $java.1$ and $asp.net.1$, the number of $\mathcal{DAS}$ remains stable across neighborhood changes. This may occur when new instances added to the neighborhood share similar attribute patterns or when the class is infrequent, resulting in fewer new instances being predicted in that class.

2. Increasing $\mathcal{DAS}$ Count: In classes like $mysql.1$ and $python.1$, the number of $\mathcal{DAS}$ increases as new instances are added with distinct attributes that influence classifier predictions. This growth reflects a dynamic adaptation in the classifiers, where additional unique attributes contribute to classifying more instances in these classes.

This analysis highlights how each classifier interprets the feature space differently, providing diverse perspectives on attribute importance across varying neighborhoods. The number of significant attributes utilized by each classifier for making predictions can differ markedly from one algorithm to another. For instance, when analyzing a neighborhood of 10 instances, both the LP and RAKEL classifiers relied on only 4% of the available attributes to accurately predict the class $mysql.1$. This observation underscores the distinct strategies employed by different classifiers in their decision-making processes. Furthermore, we have identified that the number of Decisive Attribute Sets ($\mathcal{DAS}$) for each class varies across different neighborhoods, which occurs in three primary ways:

1. **Decreasing Percentage of Significant Attributes**: In some instances, certain attributes lose their significance in class predictions due to the introduction of new instances. For example, the attributes previously deemed essential for predicting the $asp.net.1$ class may no longer hold the same predictive power in the expanded neighborhood context.

2. **Constant Percentage of Significant Attributes**: This situation arises when newly added instances share the same attributes as those predicted in the previous neighborhood, or when the class in question is infrequent, resulting in no new instances classified within that category. This stability is evident in classes like $java.1$ and $asp.net.1$, where the attribute significance remains unchanged despite neighborhood modifications.

3. **Increasing Percentage of Significant Attributes**: When new instances are introduced with distinct attributes that help the classifiers predict a given class, the percentage of significant attributes can increase. This phenomenon is observable in classes such as $mysql.1$, $java.1$, and $python.1$, where the addition of new information enhances the classifier's ability to identify and categorize instances.

This variability in the percentage of significant attributes across classifiers and neighborhoods highlights the adaptive nature of the evaluated algorithms and their reliance on the feature space context for effective predictions.

We randomly selected 10 instances to further investigate our experimental study. For each instance, we calculated the number of Decisive Attribute Sets, denoted as $\mathcal{DAS}$, as well as the percentage of Significant Attributes per class. Table 8 and Table 9 present the average number of $\mathcal{DAS}$ and the percentage of Significant Attributes for these 10 instances across different classes, with neighborhood sizes ranging from 10 to 100. In addition, we measured the execution time (in seconds) for processing each neighborhood size.

From the experimental results, we observe that the classes `asp.net.1` and `java.1` are affected by a significant class imbalance, which explains why they exhibit a low number of Decisive Attribute Sets ($\mathcal{DAS}$) and significant attributes ($\mathcal{SSA}$). This issue arises because these classes are rarely predicted by any of the three classifiers across the entire dataset. As a result, the number of $\mathcal{DAS}$ and the percentage of $\mathcal{SSA}$ do not vary substantially when moving between different neighborhood sizes. For some neighborhood sizes, these metrics are equal to zero, indicating that the classifier did not predict these classes at all. For instance, both the LP and RAKEL classifiers fail to predict the `java.1` class for small neighborhood sizes (10, 20, and 30).

In contrast, for the other classes, such as `mysql.1`, `php.1`, and `python.1`, we observe more variability in the number of $\mathcal{DAS}$ and $\mathcal{SSA}$. These metrics vary with both the number of instances predicted by the classifiers and the new attributes they use for prediction. The results indicate that as the classifier predicts more instances, the number of decisive attributes and the percentage of significant attributes tend to increase, reflecting the evolving attribute patterns associated with the class.

For example, the LP classifier used on average 16.91 $\mathcal{DAS}$ and 12.82% $\mathcal{SSA}$ to predict the `php.1` class with a neighborhood size of 90. In comparison, the ML-KNN classifier used 22.61 $\mathcal{DAS}$ and 15.91% $\mathcal{SSA}$, while the RAKEL classifier used 17.05 $\mathcal{DAS}$ and 13.1% $\mathcal{SSA}$. These differences reflect how each classifier selects and utilizes attributes to predict the classes in the dataset.

The observed differences in the number of decisive attributes and significant features across the classifiers can be attributed to the distinct approaches they employ to address multi-label classification. The LP classifier employs a *problem transformation* technique, which converts the multi-label problem into a multi-class task. It treats each combination of labels from the training set as a separate class and learns a multi-class classifier accordingly.

The ML-KNN classifier, on the other hand, follows a *problem adaptation* approach. It extends the standard KNN algorithm by integrating *Bayesian inference*,allowing the classifier to estimate both prior and posterior probabilities of each label based on the training examples.

Table 7: Significant attributes $\mathcal{SSA}$ percentage per class (for an instance). Experiments are done with 10, 20, ..., 100 neighbors using LP, ML-KNN, and RAKEL classifiers.

Neighbors	Classifiers	Asp.net.1	Java.1	Mysql.1	Php.1	Python.1
10	LP	2	0	4	0	0
	ML-KNN	1	1	10	0	6
	RAKEL	2	0	4	0	0
20	LP	1	0	11	0	7
	ML-KNN	1	1	7	0	3
	RAKEL	1	0	11	0	7
30	LP	1	1	11	1	7
	ML-KNN	1	1	11	1	6
	RAKEL	1	1	11	1	7
40	LP	1	1	14	3	8
	ML-KNN	1	1	15	5	7
	RAKEL	1	1	14	3	9
50	LP	1	1	19	8	10
	ML-KNN	1	1	20	10	9
	RAKEL	1	1	19	8	11
60	LP	1	1	23	12	11
	ML-KNN	1	1	22	12	9
	RAKEL	1	1	23	12	13
70	LP	1	1	27	11	15
	ML-KNN	1	1	25	11	13
	RAKEL	1	1	27	11	18
80	LP	1	1	28	11	16
	ML-KNN	1	1	26	11	13
	RAKEL	1	1	28	11	19
90	LP	1	1	28	11	16
	ML-KNN	1	1	26	12	13
	RAKEL	1	1	28	11	19
100	LP	1	1	32	15	16
	ML-KNN	1	1	32	16	15
	RAKEL	1	1	32	15	19

The RAKEL classifier builds on the LP approach but introduces an improvement by using *multiple multi-class classifiers*, each trained on a randomly selected subset of labels. This method enhances the flexibility and adaptability of the classifier, especially for complex multi-label datasets.

Interestingly, when considering five labels, we found that the results produced by the LP and RAKEL classifiers were very similar. This suggests that the additional complexity introduced by RAKEL does not necessarily result in better performance across all scenarios, particularly when the number of labels is relatively small.

7.3 Expermients on yelp and TMC2007-500 datasets

In addition to the Stack Overflow dataset, we also experimented with other datasets, specifically Yelp and TMC2007-500. For these datasets, we focused on neighborhood sizes of 20, 50, and 100. This choice allowed us to capture key insights without overloading the analysis with too many details, making it easier to compare classifier performance across a manageable range. By using these particular neighborhood sizes, we aimed to strike a balance between computational efficiency and capturing enough variation to reveal meaningful patterns in the data.

Yelp dataset

The dataset consists of user-generated reviews and ratings collected from Yelp. The dataset includes more than 10,000 reviews, providing a robust basis for analysis. There are 671 features and 5 binary labels which are:

- IsFoodGood: refers to the food,
- IsServiceGood: refers to the service,
- IsAmbianceGood: the look and feel of the restaurant,
- IsDealsGood: are there good deals,
- IsPriceGood: the worthiness and the value for money perceived by the reviewer.

Table 8: Average number of Decisive Attribute Sets ($\mathcal{DAS}$) per class (for 10 instances). Experiments are done with 10, 20, ..., 100 neighbors using LP, ML-KNN, and RAKEL classifiers.

Neighbors	Classifiers	Asp.net.1	Java.1	Mysql.1	Php.1	Python.1	Exectime
10	LP	0,09	0,00	1,09	0,82	0,45	0,015
	ML-KNN	0,09	0,09	1,73	1,36	0,27	0,015
	RAKEL	0,05	0	2	1,35	1,35	0,015
20	LP	0,09	0,00	3,36	1,73	1,64	0,031
	ML-KNN	0,09	0,09	3,36	2,73	0,45	0,016
	RAKEL	0,05	0	4,25	2,65	2,70	0,047
30	LP	0,09	0,09	6,09	3,36	2,55	0,046
	ML-KNN	0,09	0,18	6,91	5,09	1,36	0,047
	RAKEL	0,05	0,05	7,15	4,45	4,70	0,047
40	LP	0,09	0,18	10,18	7,00	4,00	0,047
	ML-KNN	0,09	0,27	10,91	8,00	2,73	0,062
	RAKEL	0,05	0,10	10,85	7,00	6,30	0,078
50	LP	0,09	0,18	15,27	9,64	6,00	0,078
	ML-KNN	0,09	0,45	15,73	12,09	3,36	0,067
	RAKEL	0,05	0,25	16,65	9,35	10,70	0,078
60	LP	0,09	0,36	18,64	10,73	7,91	0,093
	ML-KNN	0,09	0,45	20,64	15,73	4,36	0,093
	RAKEL	0,05	0,35	18,00	10,60	13,70	0,110
70	LP	0,09	0,45	24,09	13,00	10,73	0,094
	ML-KNN	0,09	0,82	24,73	16,09	5,91	0,110
	RAKEL	0,05	0,40	25,50	12,75	18,20	0,110
80	LP	0,09	0,73	28,55	14,73	13,18	0,140
	ML-KNN	0,09	0,91	30,27	20,64	8,09	0,140
	RAKEL	0,05	0,70	30,10	14,60	20,80	0,160
90	LP	0,09	0,91	35,73	16,91	17,36	0,170
	ML-KNN	0,09	1,00	34,36	22,64	11,09	0,170
	RAKEL	0,05	0,90	37,85	17,05	26,10	0,200
100	LP	0,09	0,91	42,55	20,00	19,36	0,230
	ML-KNN	0,09	1,00	43,45	25,64	13,18	0,210
	RAKEL	0,05	0,90	44,45	18,70	29,15	0,230

Table 10 and Table 11 show respectively the average number of $\mathcal{DAS}$ and average percentage of significant attributes $\mathcal{SSA}$ for the LP, ML-KNN and RAKEL algorithms, respectively. These results are based on 10 randomly chosen instances with neighborhoods consisting of 20, 50, and 100 neighbors.

The same findings seen on the Stack Overflow dataset also apply to the Yelp dataset. This is likely because both datasets have a reduced number of classes, which is equal to 5. However, the number of explanations ($\mathcal{DAS}$ and significant attributes $\mathcal{SSA}$) is greater with the Yelp dataset, something that is expected since the number of attributes in this dataset is greater (671 compared to 100 attributes in the Stack Overflow dataset).

TMC2007-500

The TMC2007-500 dataset is a reduced version of the Aviation Safety Reporting System (ASRS) dataset, consisting of 28,596 flight safety narratives reported by crew members. It consists of 500 features and 22 categories, each representing a distinct issue encountered during flights. This subset was generated by applying feature selection to retain the most relevant 500 terms from the original TMC2007 dataset.

As for the two previous datasets, we conducted the experiments on 10 randomly chosen instances with neighborhoods consisting of 20, 50, and 100 instances. The average results, in terms of $\mathcal{DAS}$ and $\mathcal{SSA}$, obtained by the three classifiers (ML-KNN, RAKEL, and LP), are presented in Table 12 and Table 13 below.

As observed with the other datasets, the number of explanations for a given classifier tends to increase as the neighborhood size increases and new attributes are introduced in the prediction process. However, we noticed that for the TMC2007-500 dataset, which contains 22 classes, the number of explanations obtained for the RAKEL classifier is higher. This means that RAKEL is the classifier that used the most attributes to make predictions, compared to the other two classifiers, LP and ML-KNN. This can be explained by the fact that RAKEL is a more complex classifier than LP and ML-KNN, and is therefore more able to deal with complex neighborhoods. RAKEL leverages multiple

Table 9: Average percentage of significant attributes $\mathcal{SSA}$ per class (for 10 instances). Experiments are done with 10, 20, ..., 100 neighbors using LP, ML-KNN, and RAKEL classifiers.

Neighbors	Classifiers	Asp.net.1	Java.1	Mysql.1	Php.1	Python.1	Exectime
10	LP	0,18	0,00	1,55	0,82	0,45	0,015
	ML-KNN	0,09	0,09	2,45	1,36	0,73	0,015
	RAKEL	0,10	0,00	1,90	1,15	1,40	0,015
20	LP	0,09	0,00	3,55	1,45	2,00	0,031
	ML-KNN	0,09	0,09	3,36	2,55	0,55	0,016
	RAKEL	0,05	0,00	4,25	2,05	2,85	0,047
30	LP	0,09	0,09	5,45	2,91	2,45	0,046
	ML-KNN	0,09	0,36	6,00	4,27	1,64	0,047
	RAKEL	0,05	0,05	6,35	3,50	4,20	0,047
40	LP	0,09	0,36	8,82	5,91	3,64	0,047
	ML-KNN	0,09	0,64	9,18	6,64	2,82	0,062
	RAKEL	0,05	0,20	9,80	6,15	6,55	0,078
50	LP	0,09	0,36	12,64	8,45	4,91	0,078
	ML-KNN	0,09	0,64	12,55	9,55	3,64	0,067
	RAKEL	0,05	0,35	13,40	8,30	8,00	0,078
60	LP	0,09	0,55	14,73	9,27	6,36	0,093
	ML-KNN	0,09	1,18	15,55	11,64	4,55	0,093
	RAKEL	0,05	0,45	15,80	9,50	9,45	0,110
70	LP	0,09	0,82	17,55	10,18	8,27	0,094
	ML-KNN	0,09	2,09	18,09	12,91	5,91	0,110
	RAKEL	0,05	0,55	18,30	10,30	10,70	0,110
80	LP	0,09	1,09	19,82	11,36	9,55	0,140
	ML-KNN	0,09	2,00	21,09	16,00	7,27	0,140
	RAKEL	0,05	0,90	20,70	11,65	12,70	0,160
90	LP	0,09	1,09	23,73	12,82	11,73	0,170
	ML-KNN	0,09	1,91	26,00	15,91	9,36	0,170
	RAKEL	0,05	1,00	23,25	13,10	14,80	0,200
100	LP	0,09	1,09	27,91	15,45	12,09	0,230
	ML-KNN	0,09	1,73	30,09	18,55	10,18	0,210
	RAKEL	0,05	1,00	28,10	15,15	15,90	0,230

Table 10: Average number of Decisive Attribute Sets ($\mathcal{DAS}$) per class (for 10 instances). Experiments are done with 20, 50, and 100 neighbors using LP, ML-KNN, and RAKEL classifiers.

Neighbors	Classifiers	Isfoodgood	Isservicegood	Isambiancegood	Isdealsgood	Ispricegood	Execution time
20	LP	15.18	9.64	7.09	0.00	6.55	0.82
	ML-KNN	17.91	7.36	3.91	0.82	1.64	0.85
	RAKEL	1.36	5.73	13.82	8.91	8.82	1.57
50	LP	38.18	25.18	19.45	0.00	14.64	1.13
	ML-KNN	43.91	22.82	15.27	1.27	3.73	0.90
	RAKEL	4.36	12.00	31.00	15.00	20.82	6.24
100	LP	75.82	47.09	39.18	0.00	21.82	2.21
	ML-KNN	87.09	43.36	38.00	2.09	7.91	1.66
	RAKEL	13.64	22.00	57.09	20.18	39.09	30.03

LP models trained on random subsets of labels, enabling it to capture a wider variety of attribute–label patterns in complex neighborhoods.

8 Comparative analysis with SOTA explainability methods

Our approach is distinguished for its ability to identify Decisive Attribute Sets DAS(c_k) with increased granularity, outperforming several existing attribute-based methods. Unlike approaches like LIME [17] and SHAP [18], which mainly focus on individual feature importance, our

Table 11: Significant attributes ($\mathcal{SSA}$) percentage per class (for an instance). Experiments are done with 20, 50, and 100 neighbors using LP, ML-KNN, and RAKEL classifiers.

Neighbors	Classifiers	Isfoodgood	Isservicegood	Isambiancegood	Isdealsgood	Ispricegood	Execution time
20	LP	9.89	7.15	5.87	0.00	3.95	0.82
	ML-KNN	10.55	7.71	6.62	0.33	3.84	0.85
	RAKEL	4.95	7.87	12.76	7.96	9.93	1.57
50	LP	17.73	10.89	10.56	0.00	5.89	1.13
	ML-KNN	22.13	14.00	13.02	1.33	5.22	0.90
	RAKEL	6.82	11.07	21.49	9.69	16.69	6.24
100	LP	34.98	20.33	18.98	0.00	9.16	2.21
	ML-KNN	42.02	22.49	20.22	1.42	9.27	1.66
	RAKEL	14.27	18.42	32.15	12.93	26.45	30.03

Table 12: Average number of Decisive Attribute Sets ($\mathcal{DAS}$) per class (for 10 instances).

Neighbors	Classifiers	1	2	3	...	21	22	Execution time
20	LP	0.14	9.00	0.29	...	0.00	0.00	25.57
	ML-KNN	0.86	8.57	0.00	...	0.00	0.00	16.58
	RAKEL	9.43	8.86	11.57	...	1.43	1.14	21.86
50	LP	1.43	23.14	1.43	...	0.00	0.00	82.89
	ML-KNN	1.29	22.14	0.00	...	0.00	0.00	19.90
	RAKEL	20.86	24.14	27.29	...	4.14	4.00	102.92
100	LP	4.86	44.29	4.57	...	0.00	0.14	257.87
	ML-KNN	2.43	43.14	0.00	...	0.86	0.43	42.15
	RAKEL	34.43	51.29	49.43	...	8.71	8.14	1200.42

method leverages reduced formal concepts to capture interactions between attributes. For example, in the case of the Yelp Reviews dataset, LIME could assign scores to attributes like "delicious: +0.8" or "friendly: +0.7" independently, but it does not link these attributes to explain the prediction as a whole. Similarly, SHAP, while providing additive contributions for each attribute, does not effectively distinguish the critical attribute combinations that lead to a given prediction. In contrast, our method identifies meaningful sets of attributes, such as "food quality", "fast service", and "friendly atmosphere", which together form complete justifications for a positive prediction.

In comparison to rule-based approaches like those proposed by Ribeiro et al. [41], which generate local rules, our method offers a significant advantage in terms of coverage. By leveraging formal concepts and their extents, we are able to generate both global and local explanations without needing to explicitly distinguish between the two. For instance, in the Stack Overflow dataset, our method identifies that "syntactic similarity in titles and the presence of specific keywords" is a universal rule for predicting duplicate questions, whereas rule-based approaches require a specific configuration for each individual instance. By capitalizing on interactions and providing justifications through groups of attributes, our approach better addresses the need for explainability, particularly in critical domains.

8.1 Dual-level interpretability (local and global)

Most state-of-the-art methods treat local and global explainability as distinct tasks, often requiring different algorithms and processing pipelines. Local explanations typically focus on understanding the prediction for a single in-

Table 13: Significant attributes ($\mathcal{SSA}$) percentage per class (for 10 instances).

Neighbors	Classifiers	1	2	3	...	21	22	Execution time
20	LP	0.19	6.07	0.64	...	0.00	0.00	25.57
	ML-KNN	0.32	3.53	0.00	...	0.00	0.00	16.58
	RAKEL	0.32	3.53	0.00	...	0.00	0.00	21.86
50	LP	1.20	11.89	1.73	...	0.00	0.00	82.89
	ML-KNN	0.30	6.46	0.00	...	0.00	0.00	19.90
	RAKEL	14.11	14.82	14.86	...	5.39	5.90	102.90
100	LP	1.65	17.09	3.27	...	0.00	0.47	257.87
	ML-KNN	0.32	13.05	0.00	...	1.67	1.03	42.15
	RAKEL	19.50	21.15	21.90	...	10.48	10.31	1200.42

stance (e.g., via perturbation or counterfactual reasoning), while global explanations provide aggregate insights over the dataset (e.g., feature importance or decision rules).

In contrast, our method offers a unified framework capable of producing both local and global insights. At the local level, we extract formal concepts from the neighborhood of a target instance, offering precise explanations of its predicted labels. At the global level, we aggregate recurring concepts across different instances to identify general patterns and relationships that consistently characterize specific classes. This dual-level interpretability simplifies the overall analysis workflow and enables consistent interpretive reasoning at multiple scales.

8.2 Computational efficiency and scalability considerations

Unlike model-agnostic methods such as SHAP and LIME, which rely on repeated evaluations of the predictive model, SHAP (especially in its kernel variant) exhibits exponential complexity with respect to the number of features due to its reliance on evaluating all possible feature subsets. Even optimized implementations like TreeSHAP [56] are limited to tree-based models and can be costly in terms of memory and computation.

LIME, on the other hand, perturbs the input instance to create synthetic samples and trains a surrogate model locally. This approach requires hundreds or even thousands of additional predictions, substantially increasing the computational burden.

Our method avoids both model access and sampling by leveraging a precomputed local neighborhood. It transforms this neighborhood into a formal context and applies symbolic reasoning via Formal Concept Analysis. Although in theory extracting concepts can be time-consuming when there are many attributes, in practice it runs efficiently because we only work with a small group of similar examples (the neighborhood) and most real datasets are sparse (not all features are active at once). Also, once the model has made its predictions, we don't need to retrain it or ask it for new predictions to generate explanations.

While our method performs well on standard multi-label datasets, we acknowledge that Formal Concept Analysis (FCA) can face scalability challenges, especially with high-dimensional datasets such as those encountered in image analysis or genomics. In theory, the number of formal concepts can grow exponentially with the number of attributes, making the extraction of the reduced concept lattice computationally expensive.

To address this, our approach incorporates a concept reduction strategy that significantly reduces the search space by retaining only the most relevant concepts. Furthermore, the method focuses exclusively on concepts directly related to the predicted labels, thereby mitigating the explosion of the concept space.

For very large and high-dimensional datasets, scalability can be further improved through additional optimizations. These include dimensionality reduction techniques such as feature embeddings from pre-trained convolutional neural networks (CNNs) for images or statistical/biological relevance filtering for genomics, as well as sampling strategies and parallel computation to distribute the workload. Combined, these optimizations enable our FCA-based explainability framework to remain computationally feasible and interpretable even in large-scale and complex domains.

8.3 Model-agnostic and transparent mechanism

A notable strength of our approach is its full independence from the architecture or internal structure of the underlying classifier. Since our method only requires labeled examples from a local neighborhood, it can be applied to any type of model, including non-differentiable and ensemble models. This stands in contrast to many modern techniques, such as gradient-based attribution or TreeSHAP [56], which require access to internal parameters or model structures.

Moreover, the output of our method is symbolic and human-interpretable. Explanations take the form of object–attribute–class triplets or concise anchor rules that align with human reasoning. This makes our method particularly suitable for domains where explanations must be reviewed and validated by experts, such as healthcare or policy decision-making.

In addition to symbolic outputs, we also introduce a dedicated metric to quantify the importance of each attribute with respect to a given class. This metric is derived from the frequency and specificity of attributes across the extracted formal concepts, providing a principled, transparent view of feature relevance. Unlike SHAP or LIME, which assign weights based on statistical perturbations or approximations, our importance scores are grounded in the structural patterns observed in the data. This hybrid integration of symbolic rules and importance measures strengthens the interpretability and utility of the explanations.

8.4 Capturing attribute interactions

A key limitation of many additive attribution methods like SHAP and LIME is their assumption of feature independence. These methods assign importance scores to features in isolation, missing potentially critical interactions between them. In multi-label classification, such interactions are especially important, as specific combinations of features may jointly determine the presence of multiple labels.

Our approach overcomes this limitation by leveraging the structure of formal concepts to capture meaningful attribute combinations. By analyzing patterns shared across instances with common labels, we identify sets of attributes that co-occur in a logically coherent way. These combinations provide deeper insight into how features interact and contribute collectively to predictions.

9 Concluding remarks

In this work, we introduced a novel explainability framework for multi-label classification based on Formal Concept Analysis. Our approach provides a unified mechanism for generating both local and global explanations, without requiring access to the internal structure of the model. By leveraging symbolic reasoning, we identify Decisive Attribute Sets that explain predictions in a human-understandable manner, capturing key interactions between features.

A major strength of our method lies in its ability to derive explanations that are both concise and meaningful, using reduced formal concepts that generalize across instances. Additionally, we proposed a class-specific importance measure that quantifies the relevance of each attribute based on its presence across formal concepts, further supporting interpretability. This integration of symbolic structure and attribute importance enhances trust and transparency, especially in domains where interpretability is critical.

The experimental results confirm the scalability and practical viability of our method, making it a promising tool for advancing explainability in multi-label learning.

As a perspective, this work could be expanded to include counterfactual (or contrastive) explanations, such as determining the minimal perturbations (modifications) on attribute values that would change the predictions to a predefined alternative, and to consider not only active attributes but also the absence of certain attributes as potentially informative signals.

A Additional experiments

In the experimental section, we provided details for the *mysql.1* class only. We provide details of the other remaining classes in the stack overflow dataset below:

1. php.1
   - $\mathcal{DAS}(php.1)$={{'database', 'storing', 'variable'}, {'php'}, {'change'}, {'login'}, {'java'}, {'creating'}}
   - $\mathcal{SSA}(php.1)$={'login', 'java', 'database', 'creating', 'change', 'storing', 'variable', 'php}

Table 14: Number of instances per $\mathcal{DAS}$ (class: php.1, classifier: LP, neighborhood $K = 50$)

Instance count	$\mathcal{DAS}$ used
2	{'php'}
1	{'database', 'storing', 'variable'}
1	{'change'}
1	{'login'}
1	{'java'}
1	{'creating'}

the LP classifier used 8 out of 100 attributes in our dataset to predict an instance in the class 'php.1'. This gives a rate of 8% of attributes used.

2. python.1:
   - $\mathcal{DAS}(python.1)$={'mysql', 'sqlalchemy', 'database', 'python', 'table', 'pymysql', 'database', 'database', 'python', 'mysql', 'position', 'can', 'python', 'mysql', 'database', 'mysql', 'sqlalchemy', 'string', 'arguments'}
   - $\mathcal{SSA}(php.1)$={'arguments', 'position', 'string', 'python', 'sqlalchemy', 'database', 'table', 'pymysql', 'mysql', 'can'}

Table 15: Number of instances per $\mathcal{DAS}$ (class: python.1, classifier: LP, neighborhood $K = 50$)

Instances count	$\mathcal{DAS}$ used
9	{'sqlalchemy'}
3	{'pymysql'}
3	{'database', 'python', 'mysql'}
3	{'mysql'}
3	{'database'}
2	{'table'}
1	{'database', 'python'}
1	{'position', 'can'}
1	{'database', 'mysql', 'sqlalchemy'}
1	{'position', 'can'}
1	{'string', 'arguments'}

We notice that the set of $\mathcal{DAS}$ the classifier used to predict the class python.1 is included in the set of $\mathcal{DAS}$ that it used to predict the class mysql.1; this is most likely explained by the fact that the two classes are correlated. Indeed, in the 50 instances close to the instance being explained, each time the classifier predicts the class python.1, the class mysql.1 is predicted too.
Out of the total number of attributes contained in our dataset, the LP classifier uses 10 of them to predict the python.1 class (this equals to 10% of attributes used).

3. Asp.Net.1:
   - $\mathcal{DAS}(asp.net.1)$ ={'aspnet'}
   - $\mathcal{SSA}(asp.net.1)$={'aspnet'}
4. java.1
   - $\mathcal{DAS}(java.1)$={'java'}
   - $\mathcal{SSA}(java.1)$={'java'}

References

[1] Herman B. (2017) The promise and peril of human evaluation for model interpretability, *arXiv preprint arXiv:1711.07414*.

[2] Crawford K. (2016) Artificial intelligence's white guy problem, *The New York Times*, 25(06).

[3] Caliskan A., Bryson J. J., Narayanan A. (2017) Semantics derived automatically from language corpora contain human-like biases, *Science*, 356(6334), pp. 183–186. American Association for the Advancement of Science.

[4] Barocas S., Selbst A. D. (2016) Big data's disparate impact, *California Law Review*, 104, pp. 671–. HeinOnline.

[5] Coglianese C., Lehr D. (2016) Regulating by robot: Administrative decision making in the machine-learning era, *Georgetown Law Journal*, 105, pp. 1147–. HeinOnline.

[6] Friedler S. A., Scheidegger C., Venkatasubramanian S., Choudhary S., Hamilton E. P., Roth D. (2019) A comparative study of fairness-enhancing interventions in machine learning, *Proceedings of the Conference on Fairness, Accountability, and Transparency*, ACM, pp. 329–338.

[7] Lipton Z. C. (2018) The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery, *Queue*, 16(3), pp. 31–57. ACM.

[8] Kulesza T., Burnett M., Wong W.-K., Stumpf S. (2015) Principles of explanatory debugging to personalize interactive machine learning, *Proceedings of the 20th International Conference on Intelligent User Interfaces*, ACM, pp. 126–137.

[9] Montavon G., Lapuschkin S., Binder A., Samek W., Müller K.-R. (2017) Explaining nonlinear classification decisions with deep Taylor decomposition, *Pattern Recognition*, 65, pp. 211–222. Elsevier.

[10] Golovin D., Solnik B., Moitra S., Kochanski G., Karro J., Sculley D. (2017) Google Vizier: A service for black-box optimization, *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, pp. 1487–1495.

[11] Rudin C. (2018) Please stop explaining black box models for high stakes decisions, *Stat*, 1050, p. 26.

[12] Guidotti R., Monreale A., Ruggieri S., Turini F., Giannotti F., Pedreschi D. (2018) A survey of methods for explaining black box models, *ACM Comput. Surveys*, 51(5), pp. 1–42.

[13] Datta A., Sen S., Zick Y. (2016) Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems, *2016 IEEE Symposium on Security and Privacy (SP)*, IEEE, pp. 598–617.

[14] Nguyen A., Dosovitskiy A., Yosinski J., Brox T., Clune J. (2016) Synthesizing the preferred inputs for neurons in neural networks via deep generator networks, *Advances in Neural Information Processing Systems*, 29.

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

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

[17] Guegan D. (2020) A note on the interpretability of machine learning algorithms, *University Ca'Foscari of Venice, Dept. of Economics Research Paper Series*, 20.

[18] Rodríguez-Pérez R., Bajorath J. (2020) Interpretation of machine learning models using Shapley values: application to compound potency and multi-target activity predictions, *Journal of Computer-Aided Molecular Design*, 34(10), pp. 1013–1026. Springer.

[19] Chen W., Yan J., Zhang B., Chen Z., Yang Q. (2007) Document transformation for multi-label feature selection in text categorization, *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, IEEE, pp. 451–456.

[20] Boutell M. R., Luo J., Shen X., Brown C. M. (2004) Learning multi-label scene classification, *Pattern Recognition*, 37(9), pp. 1757–1771. Elsevier.

[21] Qi G.-J., Hua X.-S., Rui Y., Tang J., Mei T., Zhang H.-J. (2007) Correlative multi-label video annotation, *Proceedings of the 15th ACM International Conference on Multimedia*, ACM, pp. 17–26.

[22] Ganter B., Wille R. (2012) *Formal Concept Analysis: Mathematical Foundations*, Springer Science & Business Media.

[23] Meddouri N., Meddouri M. (2008) Classification methods based on formal concept analysis, *Concept Lattices and Their Applications (CLA'08)*, Palacky University, Olomouc.

[24] Missaoui R., Emamirad K. (2017) Lattice Miner: a formal concept analysis tool, *14th International Conference on Formal Concept Analysis*, pp. 91.

[25] Kuznetsov S. O., Makhazhanov N., Ushakov M. (2017) On neural network architecture based on concept lattices, *International Symposium on Methodologies for Intelligent Systems*, Springer, pp. 653–663.

[26] Ganter B. (2010) Two basic algorithms in concept analysis, *International Conference on Formal Concept Analysis*, Springer, pp. 312–340.

[27] Saarela M., Jauhiainen S. (2021) Comparison of feature importance measures as explanations for classification models, *SN Applied Sciences*, 3(2), pp. 1–12. Springer.

[28] Casalicchio G., Molnar C., Bischl B. (2018) Visualizing the feature importance for black box models, *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, pp. 655–670.

[29] Lapuschkin S., Wäldchen S., Binder A., Montavon G., Samek W., Müller K.-R. (2019) Unmasking Clever Hans predictors and assessing what machines really learn, *Nature Communications*, 10(1), pp. 1–8. Nature Publishing Group.

[30] Saarela M., Kärkkäinen T. (2020) Can we automate expert-based journal rankings? Analysis of the Finnish publication indicator, *Journal of Informetrics*, 14(2), 101008. Elsevier.

[31] Weng S. F., Reps J., Kai J., Garibaldi J. M., Qureshi N. (2017) Can machine-learning improve cardiovascular risk prediction using routine clinical data?, *PLoS ONE*, 12(4), e0174944. Public Library of Science.

[32] Fisher A., Rudin C., Dominici F. (2019) All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously, *Journal of Machine Learning Research*, 20(177), pp. 1–81.

[33] Darwiche A., Hirth A. (2022) On the (Complete) Reasons Behind Decisions, *Journal of Logic, Language and Information*, pp. 1–26. Springer.

[34] Aiolli F., Sperduti A. (2010) A preference optimization based unifying framework for supervised learning problems, *Preference Learning*, Springer, pp. 19–42.

[35] Gnatyshak D., Ignatov D. I., Kuznetsov S. O. (2013) From Triadic FCA to Triclustering: Experimental comparison of some triclustering algorithms, *CLA*, 1062, pp. 249–260. Citeseer.

[36] Rudolph S., Săcărea C., Troancă D. (2015) Towards a navigation paradigm for triadic concepts, *Formal Concept Analysis: 13th International Conference, ICFCA 2015, Nerja, Spain, June 23–26, 2015, Proceedings 13*, Springer, pp. 252–267.

[37] Li Y., Zhang J., Zhang C., Zhang X. (2017) Fuzzy Formal Concept Analysis for multi-label classification, *International Journal of Approximate Reasoning*, 88, pp. 157–172.

[38] Zhang L., Zhang D., Song S. (2020) Deep learning with formal concept analysis for multi-label image classification, *Pattern Recognition*, 98, 107038.

[39] Wang Q., Liu J., Zhou J., Ye Y. (2018) Exploiting label correlations for multi-label image classification via concept factorization, *IEEE Transactions on Image Processing*, 27(8), pp. 3903–3916.

[40] Chen H., Li W., Chen L., Yu Y. (2019) Fuzzy Formal Concept Analysis for multi-label classification with uncertainty, *Knowledge-Based Systems*, 162, pp. 237–247.

[41] Ribeiro M. T., Singh S., Guestrin C. (2018) Anchors: High-precision model-agnostic explanations, *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

[42] Zhang M.-L., Li Y.-K., Liu X.-Y., Geng X. (2018) Binary relevance for multi-label learning: an overview, *Frontiers of Computer Science*, 12, pp. 191–202. Springer.

[43] Ignatov D. I., Kwuida L. (2022) On Shapley value interpretability in concept-based learning with formal concept analysis, *Annals of Mathematics and Artificial Intelligence*, 90(11), pp. 1197–1222. Springer.

[44] Kwuida L., Ignatov D. I. (2021) On interpretability and similarity in concept-based machine learning, *Analysis of Images, Social Networks and Texts: 9th International Conference, AIST 2020, Skolkovo, Moscow, Russia, October 15–16, 2020, Revised Selected Papers 9*, Springer, pp. 28–54.

[45] Sangroya A., Anantaram C., Rawat M., Rastogi M. (2019) Using formal concept analysis to explain black box deep learning classification models, *FCA4AI @ IJCAI*, pp. 19–26.

[46] Carvalho D. V., Pereira E. M., Cardoso J. S. (2019) Machine learning interpretability: A survey on methods and metrics, *Electronics*, 8(8), 832. MDPI.

[47] Sajnani H., Saini V., Kumar K., Gabrielova E., Choudary P., Lopes C. (2013) The Yelp dataset challenge – Multilabel classification of Yelp reviews into relevant categories, available at: `https://www.ics.uci.edu/~vpsaini/`.

[48] Srivastava A. N., Zane-Ulman B. (2005) Discovering recurring anomalies in text reports regarding complex space systems, *Aerospace Conference*, IEEE, pp. 3853–3862.

[49] Ciaperoni M., Xiao H., Gionis A. (2023) Concise and interpretable multi-label rule sets, *Knowledge and Information Systems*. Springer.

[50] Ayad C. W., Bonnier T., Bosch B., Read J. (2022) Shapley chains: Extending Shapley values to classifier chains, *International Conference on Discovery Science*, Springer, pp. 541–555.

[51] Hemal M. M., Saha S. (2025) Explainable deep learning-based meta-classifier approach for multi-label classification of retinal diseases, *Array*, 17, 100402. Elsevier.

[52] Mylonas N., Mollas I., Bassiliades N., Tsoumakas G. (2022) Local multi-label explanations for random forest, *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, pp. 369–384.

[53] Hassija V., Chamola V., Mahapatra A., Singal A., Goel D., Huang K., Scardapane S., Spinelli I., Mahmud M., Hussain A. (2024) Interpreting black-box models: a review on explainable artificial intelligence, *Cognitive Computation*, 16(1), pp. 45–74. Springer.

[54] Saeed W., Omlin C. (2023) Explainable AI (XAI): A systematic meta-survey of current challenges and future opportunities, *Knowledge-Based Systems*, 263, 110273. Elsevier.

[55] Saranya A., Subhashini R. (2023) A systematic review of explainable artificial intelligence models and applications: Recent developments and future trends, *Decision Analytics Journal*, 7, 100230. Elsevier.

[56] Lundberg S. M., Erion G., Chen H., DeGrave A., Prutkin J. M., Nair B., Katz R., Himmelfarb J., Bansal N., Lee S.-I. (2020) From local explanations to global understanding with explainable AI for trees, *Nature Machine Intelligence*, 2, pp. 56–67. Nature Publishing Group.

[57] Tsoumakas G., Vlahavas I. (2007) Random k-Labelsets: An ensemble method for multilabel classification, *Machine Learning: ECML 2007*, Springer, Berlin/Heidelberg, pp. 406–417.