Clustering with Stable Pattern Concepts

Egor Dudyrev[1,2,*,†], Mariia Zueva[2,†], Sergei O. Kuznetsov[2,†] and Amedeo Napoli[1,†]

[1]Université de Lorraine, CNRS, LORIA, F-54000 Nancy, France
[2]HSE University, 20 Myasnitskaya St, Moscow, 101000, Russian Federation

Abstract

Clustering aims at finding disjoint groups of similar objects in data and is one major task in Machine Learning. It is also gaining more attention in Formal Concept Analysis community in these last years. This paper proposes an original approach to the clustering of complex data based on Formal Concept Analysis (FCA) and Pattern Structures. Stable concepts are considered as cluster candidates and the SOFIA algorithm is used to discover the set of stable concepts in linear time. Then an algorithm inspired by a rare itemset mining algorithm is designed to build a clustering with good properties, i.e., high internal cohesion within a cluster and high external separation between the clusters. Some interestingness measures allowing us to choose the best clustering are discussed. Finally the present approach is compared to some other well-known algorithms such as KMeans, DBScan, and Optic.

Keywords

Formal Concept Analysis, Pattern Structures, Clustering, Rare Itemset Mining

1. Introduction

Clustering aims at finding disjoint groups of similar objects in data and is one major task in Machine Learning [1, 2]. Although the relations between clustering and Formal Concept Analysis (FCA) are known and studied since a long time [3], clustering started gaining a new interest in the FCA community in the last years [4, 5]. Besides that, it should be noticed that Conceptual Clustering [6] and Biclustering [7, 8] have always attracted attention in FCA community.

FCA can be considered as a powerful mathematical framework in data analysis and classification [9]. Thus relations between FCA and clustering are worth to study. However, FCA faces three main problems when applied to clustering. Firstly, plain FCA only considers so called Formal Contexts based on binary datasets while most of the data are either numerical or of more complex nature. Secondly, without additional constraints, concept lattices can be exponential in the size of data (formal contexts) which makes plain FCA algorithms not applicable to big data. Thirdly, FCA concepts are organized in a concept lattice and are overlapping, while clustering is based on a partition into non-overlapping clusters. In this paper we propose an original approach to overcome these three problems: (1) we use Pattern Structures to extend FCA to deal with (almost) any kind of complex descriptions, (2) we use the SOFIA algorithm to discover a limited set of cluster candidates in linear time, and (3) we propose an algorithm to select non-overlapping clusters from the set of given cluster candidates based on Rare Itemset Mining.

Pattern Structures are used for clustering in [4], where authors are considering Pattern Structures adapted to numerical and sequential data. The present paper studies clustering of tabular data of any type, where every column is represented by an arbitrary pattern structure, making the present approach more versatile and more universal.

FCA4AI 2024: The 12th International Workshop "What can FCA do for Artificial Intelligence?", October 19 2024, Santiago de Compostela, Spain.
*Corresponding author.
†These authors contributed equally.
egor.dudyrev@loria.fr (E. Dudyrev); m.zueva@hse.ru (M. Zueva); skuznetsov@hse.ru (S. O. Kuznetsov); amedeo.napoli@loria.fr (A. Napoli)
https://egordudyrev.github.io/ (E. Dudyrev)
0000-0002-2144-3308 (E. Dudyrev); 0009-0000-6332-9936 (M. Zueva); 0000-0003-3284-9001 (S. O. Kuznetsov); 0000-0001-5236-9561 (A. Napoli)

The idea of concept stability in FCA was first introduced in [10] and then refined in [11] giving rise to Δ-stability. Roughly speaking, the Δ-stability of a concept shows how many objects the concept will lose when making its description more precise (recall that a concept is composed of a set of objects and a set of attributes materializing objects' common description). The use of concept stability for selecting concepts adapted to clustering is studied in [12]. However, these authors are using concept stability to select interesting concepts from the whole set of concepts which may be of exponential size. By contrast, in our approach we make use of the Sofia algorithm [13, 14], to select the stable concepts in linear time, without requiring to construct the whole set of concepts.

The problem of avoiding overlapping clusters when covering the whole data is addressed in various ways. For example, the authors of [6] are trying to discover similar sublattices of concepts w.r.t. a predefined similarity measure. Thus the latter approach is closer to "Conceptual Clustering" rather than to clustering of objects. The authors of [15] are solving a biclustering problem, which is more specific than clustering, and, firstly they are discovering a set of non-overlapping concepts that covers most of the data, and then they are adding missing objects to the discovered biclusters until the whole data is covered. By contrast, the authors of [4] are considering overlapping concepts as clusters, and then they go through every object in the overlap and assign it to one cluster to which it "mostly" belongs.

From our side, we think that the overlap between clusters is a natural phenomenon as there are many things in our world which cannot be strictly attached to only one single concept[1]. Thus, by contrast, we propose to build clusters with the smallest possible overlap, and then to draw the attention of the analyst to these overlapping objects.

2. Concepts as Clusters

2.1. A Bit of Formal Concept Analysis Terminology

Formal Concept Analysis [9] is a mathematical formalism based on lattice theory and aimed at data analysis and classification. In FCA, data are represented with a **formal context** (G, M, I) where G is the set of objects, M is the set of attributes, and $I \subseteq G \times M$ is the binary relation between objects and attributes. A formal context or more simply context can be represented as a binary table where rows stand for objects, columns for attributes, and a cross is lying in a cell when the corresponding object has the corresponding attribute.

Given a formal context (G, M, I), we define two **derivation operations** denoted as $(\cdot)'$ ("prime"): given a set of objects $A \subseteq G$, the first operation returns $A' \subseteq M$, i.e., the set of attributes common to all objects in A, while, given a set of attributes $B \subseteq M$ the second operation returns $B' \subseteq G$, i.e., the set of all objects having all attributes in B. More formally:

$$A' = \{m \in M \mid \forall g \in A, (g, m) \in I\}, \quad B' = \{g \in G \mid \forall m \in B, (g, m) \in I\}. \tag{1}$$

For the sake of simplicity, we denote the description on a single object $g \in G$ as g' rather than $\{g\}'$, while we denote the objects described by a single attribute $m \in M$ as m', rather than $\{m\}'$.

A **formal concept** (A, B) is a pair where the set of objects A and the set of attributes B verify $A' = B$ and $B' = A$. In concept (A, B), the set of objects A is called the **extent** and the set of attributes B is called the **intent**. Moreover, concepts can be organized into a concept lattice thanks to the subsumption relation –a partial ordering– where a concept (A_1, B_1) is subsumed by a concept (A_2, B_2) iff $A_1 \subseteq A_2$ or dually $B_2 \subseteq B_1$

A subset of attributes $D \subseteq M$ is called a **minimal generator** of concept (A, B) when it is a minimal subset of attributes, whose extent is A. In other words, removing any attribute $m \in D$ from description D will change its extent, i.e., $\forall m \in D, (D \setminus \{m\})' \neq D'$.

Finally, the **support** of any description $D \subseteq M$ is given by the cardinality of the set of objects having D as description, i.e., $\mathtt{supp}(D) = |D'|$.

[1] An interesting example is given by "Pheasant Island", that belongs either to France or to Spain depending on the time of the year!

2.2. Formal Concepts as Clusters

Clustering is generally defined as the problem of discovering a set of disjoint clusters that cover all the data, such that objects belonging to the same cluster are more similar than objects belonging to different clusters. The choice of a similarity measure depends on a the type of data and the task at hand. For example, considering numerical data, in K-means clustering (see for example [16]) every object is described by a vector of real numbers and the similarity between objects is in the inverse proportion to the Euclidean distance between the object descriptions. In DBScan clustering (see again [16]) the similarity between objects is based on the amount of close common neighbours in the Euclidean space.

In our framework, two objects $g_1, g_2 \in G$ are described by the corresponding sets of attributes –aka itemsets– $g_1', g_2' \subseteq M$. A natural way to define the similarity between two objects is provided by the Jaccard similarity coefficient [16] between the descriptions:

$$\texttt{sim}(g_1, g_2) \coloneqq J(g_1', g_2') = \frac{|g_1' \cap g_2'|}{|g_1' \cup g_2'|}. \tag{2}$$

Using equation 2 the clustering task can be defined as follows, where $\wp(G)$ is the powerset of the set of objects G:

$$\begin{aligned} &\text{Discover a set of clusters } \mathcal{C} \subseteq \wp(G), \text{ such that:} \\ &\bigcup_{C_i \in \mathcal{C}} C_i = G \\ &\forall\, C_i,\, C_j \in \mathcal{C}, C_i \cap C_j = \emptyset \\ &\forall\, g,\, g_i \in C_i, g_j \in C_j : \texttt{sim}(g, g_i) \gg \texttt{sim}(g, g_j) \end{aligned} \tag{3}$$

Now let us consider a formal concept (A, B) and the similarity between two objects $g_1, g_2 \in A$:

Proposition 2.1. *Given a formal concept (A, B), the similarity between any pair of objects from extent A is lower-bounded by the length of the concept intent $|B|$:*

$$\texttt{sim}(g_1, g_2) \geq \frac{|B|}{|M|} \tag{4}$$

Proof. Since g_1 and g_2 belong to concept (A, B), the concept's intent B is included in their common description $g_1' \cap g_2'$. Meanwhile the union of the descriptions $g_1' \cup g_2'$ cannot be larger than the maximal description M, and then $g_1' \cup g_2' \subseteq M$. Thus, the following formulas hold true:

$$\texttt{sim}(g_1, g_2) = \frac{|g_1' \cap g_2'|}{|g_1' \cup g_2'|} \geq \frac{|B|}{|g_1' \cup g_2'|} \geq \frac{|B|}{|M|}.$$

□

Therefore, a formal concept (A, B) can be considered as a cluster of objects A that are at least $|B|/|M|$ similar. Following this reformulation, the objective of clustering is to discover a set of concepts with large intents but tiny or no overlapping between the extents of concepts.

3. Clustering pipeline

The clustering pipeline proposed in this paper is shown in Figure 1. Below we discuss the different FCA techniques allowing us to efficiently build an optimal clustering.

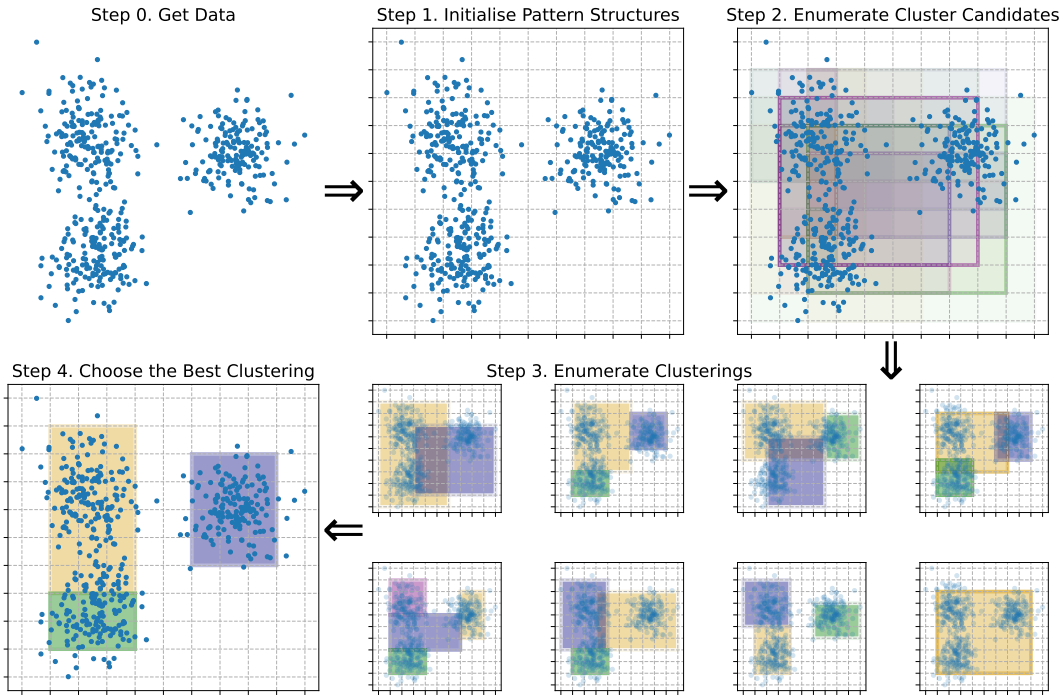

Figure 1: The pipeline proposed for building a clustering pipeline based on FCA techniques.

3.1. Step 1. Initializing the Pattern Structure

While plain FCA works with contexts representing binary datasets, we rely in the present framework on Pattern Structures, an extension of FCA allowing us to deal with many types of complex data. In the following, we focus on Interval and Cartesian Pattern Structures to take into account multidimensional numerical data. Below we recall the definitions and techniques of Pattern Structures.

Recall that a Formal Context is a triple (G, M, I) where G is a set of objects, M is a set of attributes, and $I \subseteq G \times M$ is a set of pairs (g, m) indicating that object g is described by attribute m. In such a formal context, the space of object descriptions $\underline{\mathbb{D}} = (\mathbb{D}, \subseteq)$ is the powerset of attributes $\mathbb{D} = \wp(M)$, ordered by inclusion $\subseteq$. It should be noticed that such description space $\underline{\mathbb{D}} = (\mathbb{D}, \subseteq)$ forms a lattice, i.e., for every pair of descriptions $D_1, D_2 \in \mathbb{D}$ there is exactly one meet and one join: $\exists! D_\wedge \in \mathbb{D}, D_1 \cap D_2 = D_\wedge$ and $\exists! D_\vee \in \mathbb{D}, D_1 \vee D_2 = D_\vee$.

The Pattern Structure formalism [17, 18, 19] generalizes plain FCA and in particular the description space $\underline{\mathbb{D}}$. The latter consists of a description set $\mathbb{D}$ equipped with operation $\sqcap$ that defines a complete meet semilattice on $\mathbb{D}$, i.e., for any pair of descriptions $D_1, D_2 \in \mathbb{D}$ there exists meet (infimum): $\exists! D_\wedge \in \mathbb{D}, D_1 \sqcap D_2 = D_\wedge$. The operation $\sqcap$ defines natural order $\sqsubseteq$: $X \sqsubseteq Y \iff X \sqcap Y = X$. One can interpret $X \sqsubseteq Y$ as "X is a less precise description than Y".

Then the description space $\underline{\mathbb{D}}$ combined with the set of objects G and a mapping $\delta : G \to \mathbb{D}$ forms a **pattern structure** $(G, \underline{\mathbb{D}}, \delta)$, that can be considered as an analogue and a generalization of a formal context (G, M, I). For any pattern description $D \in \mathbb{D}$, one can define the pattern extent $D^\diamond$, and for any subset of objects $A \subseteq G$, one can define the pattern intent $A^\diamond$:

$$D^\diamond = \{g \in G \mid D \sqsubseteq \delta(g)\}, \quad A^\diamond = \sqcap\{\delta(g) \mid g \in A\}. \tag{5}$$

A pair of corresponding pattern extent A and pattern intent D forms a **pattern concept**: (A, D), where $A^\diamond = D, D^\diamond = A$.

This paper focuses on Interval and Cartesian Pattern Structures for modelling multidimensional numerical data. The **Interval Pattern Structure** works with the description space $\underline{\mathbb{D}}_{\text{int}}$ of intervals bounded by real numbers $\mathbb{D}_{\text{int}}$ ordered by interval subsumption $\sqsubseteq_{\text{int}}$:

$$\mathbb{D}_{\text{int}} = \{[l, r] \mid l, r \in \mathbb{R}, l \leq r\}, \text{ and } \forall [l_1, r_1], [l_2, r_2] \in \mathbb{D}_{\text{int}}, [l_1, r_1] \sqsubseteq_{\text{int}} [l_2, r_2] \iff [l_1, r_1] \supseteq [l_2, r_2].$$

The **Cartesian Pattern Structure** allows to combine various description spaces $\underline{\mathbb{D}}_1, \underline{\mathbb{D}}_2, \ldots, \underline{\mathbb{D}}_n$ in a single description space $\underline{\mathbb{D}}_\times = (\mathbb{D}_\times, \sqsubseteq_\times)$ such that:

$$\mathbb{D}_\times = \bigtimes_{1 \leq i \leq n} \mathbb{D}_i \text{ and } \forall D, E \in \mathbb{D}_\times, D \sqsubseteq_\times E \iff \bigwedge_{1 \leq i \leq n} D_i \sqsubseteq_i E_i.$$

In this paper, every column in a numerical dataset is processed thanks to a distinct Interval Pattern Structure $(G, \underline{\mathbb{D}}_i, \delta_i)$, and the whole dataset is processed thanks to a Cartesian Pattern Structure $(G, \underline{\mathbb{D}}_\times, \delta_\times)$ built on top of base Interval Pattern Structures.

We also provide the "support function" `supp` to compute either the number of objects described by a binary description $B \subseteq M$, or the number of objects described by a pattern description $D \in \mathbb{D}$:

$$\texttt{supp}(B) = |B'|, \forall B \subseteq M, \quad \texttt{supp}(D) = |D^\diamond|, \forall D \in \mathbb{D} \tag{6}$$

This allows us to highlight the similarities when using either FCA or Pattern Structures for the clustering task.

Although Interval Pattern Structures may work with an infinite space of intervals $\mathbb{D}$, we restrict each pattern structure to only deal with 11 evenly-spaced interval borders, i.e., $V \subseteq \mathbb{R}, |V| = 11$, $\mathbb{D} = \{[l, r] \mid l, r \in V, l \leq r\}$, assuming that every object description lies inside the largest interval $[\min(V), \max(V)]$. This restriction allows us to reduce the computational time and to improve the stability of descriptions, as this is discussed in the next section.

3.2. Step 2. Enumerating Cluster Candidates

Previously, we have defined the Jaccard similarity measure between two objects in a formal context. However, it is much less straightforward to define a similarity measure in a Pattern Structure. For example, considering the two descriptions of cities: "population from 10k to 100k people, in East Asia" and "population from 100k to 1M people, in France". Discovering which cities are the more similar depends on the arbitrary similarity function defined for each pattern dimension, e.g., population and geographical location, and on the arbitrary way to aggregate these similarity functions into a single similarity measure. Below we describe how to use stable concepts to mimic the similarity for any type of descriptions.

Formal Concept stability is defined in [10] as the percentage of subsets in a concept extent having a common description B:

$$\texttt{stab}(A, B) \coloneqq \frac{|\{A_2 \subseteq A \mid A_2' = B\}|}{2^{|A|}}. \tag{7}$$

However, due to its exponential nature, stability is hard to compute in practice. This is why Δ-stability was introduced in [11] as a linear-time upper bound of concept stability:

$$\Delta\texttt{stab}(A, B) \coloneqq |A| - \max_{\substack{B_2 \subseteq M \\ \text{s.t. } B \subset B_2}} \texttt{supp}(B_2). \tag{8}$$

Delta-stability can also be adapted to a pattern concept (A, D) whose description belongs to a description space $\mathbb{D}$:

$$\Delta\texttt{stab}(A, D) \coloneqq |A| - \max_{\substack{D_2 \in \mathbb{D} \\ \text{s.t. } D \sqsubset D_2}} \texttt{supp}(D_2). \tag{9}$$

In general terms, the value of Δ-stability can be interpreted as "how many objects from A one will lose when making description D just a bit more precise". Then, for a concept (A, D) with a high Δ-stability, even though the exact similarity between objects in A is unmeasurable, one knows that any cluster of more similar objects would contain significantly fewer objects.

Another useful characteristic of stable concepts is that there exist efficient algorithms such as SOFIA [13] and gSOFIA [14] that can be used to directly mine only stable concepts without computing the excessively large amount of non-stable concepts.

3.3. Step 3. Enumerating Clustering Candidates

The previous sections showed how one can mine individual clusters. This sections discusses how one can combine individual clusters into clusterings (i.e. subsets of clusters). Specifically, we focus on enumerating clusterings with two properties: a clustering should cover most of the objects in the data and clusters should not overlap too much. Below we present an algorithm enumerating clustering candidates satisfying these two properties.

More formally, let us first consider a set of clusters candidates $\mathbb{C} \subseteq \wp(G)$, where every cluster $C \in \mathbb{C}$ is a closed set of objects, i.e., $C'' = C$. Then a **clustering** is any subset of clusters $\mathcal{C} \subseteq \mathbb{C}$. Let us define **coverage** $\texttt{cov}(\mathcal{C})$ of clustering $\mathcal{C}$ as the number of objects belonging to at least one cluster in $\mathcal{C}$: $\texttt{cov}(\mathcal{C}) = |\bigcup_{C_i \in \mathcal{C}} C_i|$. Then, a clustering $\mathcal{C}$ is called **broad** if it covers more than θ_{cov} objects: $\texttt{cov}(\mathcal{C}) > \theta_{cov}$. A clustering $\mathcal{C}$ is called **minimal broad** clustering if it is a broad clustering, and all its proper subsets are not broad clusterings: $\texttt{cov}(\mathcal{C}) > \theta_{cov}$ and $\forall \mathcal{C}_2 \subset \mathcal{C}, \texttt{cov}(\mathcal{C}_2) \leq \theta_{cov}$. A clustering $\mathcal{C}$ is called θ_{ol}-**non-overlapping** if every pair of clusters overlaps for at most θ_{ol} objects: $|C_i \cap C_j| \leq \theta_{ol}, \forall\, C_i,\, C_j \in \mathcal{C}$. Then, our task consists in enumerating minimal broad non-overlapping clusterings built from the set of clusters $\mathbb{C}$.

The latter problem of discovering minimal broad non-overlapping clusterings is far from being simple, as it can even be related to the famous Set Covering Optimisation Problem. However, a satisfactory solution can be found when the problem is related to the "Rare Itemset Mining" problem [20], which was formerly addressed in the pattern mining and FCA communities. Rare Itemset Mining focuses on discovering **minimal rare itemsets**, that are minimal subsets of attributes $D \subseteq M$ of a formal context (G, M, I) whose support is below a given threshold $\theta_{\min}$: $D \subseteq M$ s.t. $\texttt{supp}(D) = |\bigcap\{m' \mid m \in D\}| < \theta_{\min}$ and $\forall m \in D, \texttt{supp}(D \setminus \{m\}) \geq \theta_{\min}$.

It can be noticed that discovering **minimal broad clusterings** –possibly overlapping– can be reduced to discovering minimal rare itemsets, as minimal broad clusterings are the minimal subsets of clusters $\mathcal{C} \subseteq \mathbb{C}$ whose coverage is above a given threshold θ_{cov}, i.e., $\mathcal{C} \subseteq \mathbb{C}$ s.t. $\texttt{cov}(\mathcal{C}) = |\bigcup C_i| > \theta_{cov}$ and $\forall C_i \in \mathcal{C}, cov(\mathcal{C} \setminus \{C_i\}) \leq \theta_{cov}$.

The relationship between discovering minimal broad clusterings and minimal rare itemsets allows us to use Rare Itemset Mining algorithms for finding minimal broad clusterings. To do so, one should search for minimal rare itemsets in the inverted clusters context $(G, \mathbb{C}, \notin)$ with minimal support threshold $\theta_{\min} = |G| - \theta_{cov}$.

Proposition 3.1. *Let us consider the "inverted cluster context" $K_{\overline{\mathbb{C}}} = (G, \mathbb{C}, \notin)$, where G is a set of objects, $\mathbb{C}$ a set of clusters, and $\notin$ the incidence relation such that $\notin = \{(g, C_i) \in G \times \mathbb{C} | g \notin C_i\}$.*

Then a subset of clusters $\mathcal{C} \subseteq \mathbb{C}$ is a minimal broad clustering in $K_{\overline{\mathbb{C}}} = (G, \mathbb{C}, \notin)$ w.r.t. the coverage threshold θ_{cov} if and only if it is a minimal rare itemset in context $K_{\overline{\mathbb{C}}}$ w.r.t. the minimal support threshold $\theta_{\min} = |G| - \theta_{cov}$.

Proof. Consider the logical statement over two literals a and b: $\overline{a} \wedge \overline{b} = \overline{a \vee b}$. Now, let a, b be attributes of an arbitrary formal context (G, M, I). An analogous property of attribute extents can be inferred: $(G \setminus a') \cap (G \setminus b') = G \setminus (a' \cup b')$.

In the inverted cluster context $K_{\overline{\mathbb{C}}} = (G, \mathbb{C}, \notin)$, every attribute is a cluster $C \in \mathbb{C}$The context is designed in such a way –thanks to the the $\notin$ relation– that the extent of every cluster-as-attribute is the complement of the cluster itself: $C' = G \setminus C$. Here the expression "cluster-as-attribute" stands for an attribute representing a cluster in the inverted context.

Thus, the extent of any subset of clusters-as-attributes $D \subseteq \mathbb{C}$ in this context contains objects described by none of the clusters: $\bigcap\{C'_i \mid C_i \in D\} = G \setminus \bigcup\{C_i \mid C_i \in D\}$. Given that $\texttt{supp}(D) = |\bigcap\{C'_i \mid C_i \in D\}|$ and $\texttt{cov}(D) = |\bigcup\{C_i \mid C_i \in D\}|$, it comes that $\texttt{supp}(D) = |G| - \texttt{cov}(D)$. This equality, in turn, gives rise to the proposition, i.e., $\texttt{supp}(D) < \theta_{\min} \iff \texttt{cov}(D) > |G| - \theta_{\min} = \theta_{cov}$. □

To the best of our knowledge, paper [21] was the first to propose the idea of representing the union of attributes extents via the intersection of the same extents in the inverted context. There, the authors

considered the unions of attributes as the intents of monotone Galois connections and used formal concepts to mimic the behaviour of linear regressions and neural networks.

As we have shown, one can reuse algorithms of Rare Itemset Mining to enumerate all broad clusterings. For example, in this work we have re-implemented the algorithm MRG-Exp (also known as Carpathia-G-Rare) proposed in [20]. However, there are two peculiarities of a clustering task that are not really considered in Rare Itemset Mining: (1) minimal rare itemsets may contain an arbitrary amount of attributes, while a clustering often contains only a few clusters, and (2) attributes in a minimal rare itemset may have highly overlapping extents, while clusters in a clustering are supposed to be disjoint. To satisfy these two requirements, we add two parameters in our implementation of MRG-Exp algorithm. Firstly, we introduce **maximal size parameter** η_{size} to only consider clusterings $\mathcal{C}$ containing at most η_{size} clusters: $|\mathcal{C}| \leq \eta_{\text{size}}$. And secondly, we add **minimal added coverage parameter** η_{cov} that defines the minimal amount of objects a cluster should add to a clustering: $\forall C \in \mathcal{C}, \texttt{cov}(\mathcal{C}) - \texttt{cov}(\mathcal{C} \setminus \{C\}) \geq \eta_{\text{cov}}$. It can be noticed that, when η_{cov} is set to 1, the condition on minimal added coverage becomes the condition on the minimality of a clustering: $\forall C \in \mathcal{C}, (\texttt{cov}(\mathcal{C}) - \texttt{cov}(\mathcal{C} \setminus \{C\}) \geq 1) \iff (\texttt{cov}(\mathcal{C}) \neq \texttt{cov}(\mathcal{C} \setminus \{C\}))$.

To summarize this section, we state that we solve the problem of enumerating minimal broad non-overlapping clusterings by relating it to the problem Rare Itemset Mining with an additional non-overlapping requirement. That is, we re-implement the MRG-Exp algorithm, while replacing all intersections of extents in the algorithm with their unions, and replacing all tests of the form "support < $\theta_{\min}$" by dual tests of the form "coverage > θ_{cov}". Finally, we reduce the search space of clusterings by specifying the restriction on the maximal size of a clustering, and by specifying the minimal added support threshold for every concept in a clustering.

3.4. Step 4. Selecting the Best Clustering

Now we know how to enumerate minimal broad non-overlapping clustering candidates. However, one can obtain multiple –sometimes, thousands of– minimal broad non-overlapping clustering candidates. Below we propose some measures for guiding the choice of the best clustering out of the possibly very large set of broad minimal non-overlapping candidates.

The main criterion for interestingness –or goodness– of a clustering $\mathcal{C} \subseteq \wp(G)$ is the **coverage** of the clustering, i.e., the number of objects covered by the clustering $\mathcal{C}$: $\texttt{cov}(\mathcal{C}) = |\bigcup C_i|$.

The second most important criterion for goodness of a clustering $\mathcal{C} \subseteq \wp(G)$ is the **overlap**, i.e., the size of the pairwise intersections of clusters in $\mathcal{C}$: $\texttt{ovlap}(\mathcal{C}) = \sum_{C_i, C_j \in \mathcal{C}} |C_i \cap C_j|$. Note that here we do not normalise the size of the overlaps by the number of pairs of concepts. The normalisation procedure we follow is explained at the end of this section.

Another criterion for differentiating two clustering candidates is to measure their **sizes**: $\texttt{size}(\mathcal{C}) = |\mathcal{C}|$. Depending on the task and the data, the analyst running the clustering might prefer clustering candidates with a specific number of clusters.

Moreover, in some cases, an analyst may prefer or penalize imbalanced clustering candidates where the sizes of the clusters in the clustering $\mathcal{C}$ are highly varying. The **imbalance** of a clustering $\mathcal{C}$ is measured as the standard deviation of the cardinalities of its clusters: $\texttt{imb}(\mathcal{C}) = \texttt{std}(\langle |C_1|, |C_2|, \ldots, |C_{|\mathcal{C}|}| \rangle)$.

An analyst may also prefer clustering candidates consisting of mostly stable concepts. Then the **stability** of a clustering $\mathcal{C}$ is measured as the average delta-stability of its concepts: $\texttt{stab}(\mathcal{C}) = \sum_{C_i \in \mathcal{C}} \Delta\texttt{stab}(C_i)/|\mathcal{C}|$.

Finally, since we study multidimensional numerical data, we will give priority to dense clusters. More precisely, in n-dimensional data, the clusters have the form of hyperrectangles, i.e., $D = \langle [l_1, r_1], [l_2, r_2], \ldots, [l_n, r_n] \rangle$. The **density** of a clustering $\mathcal{C}$ is defined as the average density of its clusters-concepts $(A, D) \in \mathcal{C}$: $\texttt{density}(\mathcal{C}) = \sum_{(A,D) \in \mathcal{C}} \texttt{density}\big((A, D)\big)$, where $\texttt{density}\big((A, D)\big) = |A| / \Pi_{j=1}^{n}(r_j - l_j)$.

In order to aggregate all measures related to a clustering in a single measure, every clustering is associated with a **reward function**, which is a weighted sum of the above measures. In addition, to

improve the interpretability of weights in the reward function, we normalize the values of each basic measure in such a way that the lowest possible value of any basic measure is 0, and the maximal possible value of any basic measure is 1, i.e., we apply MinMax scaling to the values of the computed basic measures.

4. Experiments and Discussion

This paper presents our first attempt in building a clustering problem based on FCA and Pattern Structures. For testing these first ideas, we have run tests over artificial and accessible datasets provided by SciKit Learn to compare the present results with various State-of-the-Art clustering algorithms. The results of the original algorithm comparison is presented on the web page https://scikit-learn.org/stable/modules/clustering.html.

We chose to compare the experiment results returned by or *FCA-based* algorithm that follows the pipeline presented above, with three well-known clustering methods. We considered (1) *K-Means* which is one of the most popular and the most simple clustering methods, (2) *DBScan* which is one of the most popular density-based clustering method, and (3) *Optics* which is one of the most versatile –while also the less time efficient– algorithm presented in SciKit Learn.

The plots on Figure 2 present the clustering obtained by 4 algorithms on 6 datasets. It can be seen that no clustering method is perfect: for example, K-Means does not work well on circular data (the top row #1), DBscan and Optics do not find all three clusters on the "blobs" data (row #5), while FCA-based algorithm works nicely on "blobs"-based data (rows #3 and #5) but fails on the other datasets.

It should be noticed that all these different clustering methods are based on different principles and processes. K-Means clustering operates over centroids of clusters in multi-dimensional data. Thus, it naturally tends to discover “blobs”-like clusters (rows #3, #5). DBScan and Optics are density-based approaches. Therefore, they tend to discover nonlinear continuous clusters (e.g. rows #1, #2, #4) but fail when the objects of two clusters are placed too close to each other. Finally, the FCA-based algorithm searches for clusters that having more the form of a hyperrectangle. Thus, the latter tends to discover “blobs”-like clusters as K-Means does.

The main disadvantage of the current FCA-based algorithm is the running time. As the results in Figure 2 show, the FCA-based approach may work up to 1780 times slower than the the slowest competitor which is Optics. Table 1 presents the running times and the sizes of the output computed at each step of the proposed pipeline. It can be seen that most of the time is spent in Step 3 of the pipeline, corresponding to the computing of minimal broad non-overlapping clustering candidates. Actually, during this step hundreds of thousands of clustering candidates are produced leading to a very high redundancy, while only a few best candidates are interesting. The minimization of the number of clustering candidates discovered during Step 3 will also reduce the time required in Step 4 of the pipeline, whose objective is the evaluation of the returned clustering candidates.

Thus, an important direction in future work is to develop a new algorithm for finding only hundreds of best broad non-overlapping clustering candidates. Meanwhile, it should be noticed that in most of the cases the total running time in Table 1 are already lying within “reasonable time slots” of tens of seconds.

One could argue that an FCA-based algorithm can also find nonlinear clusters, as in rows #1, #2, and #4, when using a polygon-based pattern structure (see [22, 23]) instead of the combination of Interval and Cartesian pattern structures. Indeed, this is also one main future work.

The source code for the experiments can be found in the Git repository https://github.com/EgorDudyrev/Paper_StablePatternClustering. The results for these experiments were obtained on a MacBook Pro with Apple M2 chip and 16 GB of RAM.

dataset	Step 2		Step 3		Step 4	total time (s)
	# stable concepts	stable concepts time (s)	# clusterings	clusterings time (s)	statistics time (s)	
noisy_circles	1 150	0.06	129 629	84.73	4.28	89.07
noisy_moons	636	0.04	99 082	15.86	3.08	18.98
varied	564	0.04	71 696	8.77	2.26	11.07
aniso	342	0.03	21 353	1.55	0.96	2.54
blobs	554	0.04	51 796	7.17	2.37	9.57
no_structure	1 139	0.05	96 914	84.18	3.19	87.42

Table 1
The time and the size of the output for every step of the proposed clustering pipeline.

5. Conclusions

In this paper we have presented an original pipeline for clustering numerical data using Formal Concept Analysis and Pattern Structures. The pipeline consists of four steps: (1) we encode the data via Interval and Cartesian Pattern Structures, (2) we find the set of stable cluster candidates thanks to the gSofia algorithm, (3) we enumerate the set of minimal broad non-overlapping clustering candidates, and (4) we select the best clustering candidates based on a set of interestingness measures. We also show that this approach outputs some reasonable clusterings when applied to artificial datasets from the SciKit Learn package, while running in a matter of seconds.

As future work we are planning to mainly improve the third step of the pipeline, by reducing the space of the clustering candidates. We will also run experiments over real-world complex datasets with numerical, categorical, and textual elements. Finally, our research raises the question of the type of clusters that can be found when using an FCA framework, i.e., how to define a pattern structure able to describe dense continuous clusters, or rotated hyperrectangles, or any polygons in multidimensional space.

6. Acknowledgments

Egor Dudyrev and Amedeo Napoli are carrying out this research work as part of the French ANR-21-CE23-0023 SmartFCA Research Project. The work by S.O. Kuznetsov in preparing this article was supported by grand 22-11-00323 of the Russian Science Foundation and carried out at the National Research University Higher School of Economics, Moscow.

7. Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] L. Kaufman, P. J. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley, 1990.
[2] B. G. Mirkin, Core Data Analysis: Summarization, Correlation, and Visualization, Second Edition, Undergraduate Topics in Computer Science, Springer, 2019.
[3] C. Carpineto, G. Romano, A Lattice Conceptual Clustering System and Its Application to Browsing Retrieval, Machine Learning 24 (1996) 95–122.
[4] S. E. Boukhetta, M. Trabelsi, Formal Concept Analysis for Trace Clustering in Process Mining, in: M. Ojeda-Aciego, K. Sauerwald, R. Jäschke (Eds.), Graph-Based Representation and Reasoning, Springer Nature Switzerland, Cham, 2023, pp. 73–88.

[5] M. Alwersh, L. Kovács, K-Means Extensions for Clustering Categorical Data on Concept Lattice, International Journal of Advanced Computer Science and Applications (IJACSA) 14 (2023).
[6] T. T. Quan, S. C. Hui, T. H. Cao, A Fuzzy FCA-based Approach to Conceptual Clustering for Automatic Generation of Concept Hierarchy on Uncertainty Data, in: CLA 2004, volume 2004, VS˘B – Technical University of Ostrava, Dept. of Computer Science, 2004, pp. 1–12.
[7] N. Juniarta, M. Couceiro, A. Napoli, A Unified Approach to Biclustering Based on Formal Concept Analysis and Interval Pattern Structure, in: P. K. Novak, T. Smuc, S. Dzeroski (Eds.), Proceedings of 22nd International Conference on Discovery Science (DS), Lecture Notes in Computer Science 11828, Springer, 2019, pp. 51–60.
[8] D. I. Ignatov, B. W. Watson, Towards a Unified Taxonomy of Biclustering Methods, 2017. doi:10.48550/arXiv.1702.05376. arXiv:1702.05376.
[9] B. Ganter, R. Wille, Formal Concept Analysis, Springer, Berlin, 1999.
[10] S. O. Kuznetsov, On stability of a formal concept, Annals of Mathematics and Artificial Intelligence 49 (2007) 101–115.
[11] A. Buzmakov, S. O. Kuznetsov, A. Napoli, Scalable Estimates of Concept Stability, in: C. V. Glodeanu, M. Kaytoue, C. Sacarea (Eds.), Formal Concept Analysis, Lecture Notes in Computer Science 8478, Springer International Publishing, Cham, 2014, pp. 157–172.
[12] J. Cigarrán, Á. Castellanos, A. García-Serrano, A step forward for Topic Detection in Twitter: An FCA-based approach, Expert Systems with Applications 57 (2016) 21–36.
[13] A. Buzmakov, S. O. Kuznetsov, A. Napoli, Fast Generation of Best Interval Patterns for Nonmonotonic Constraints, in: A. Appice, P. P. Rodrigues, V. Santos Costa, J. Gama, A. Jorge, C. Soares (Eds.), Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD), Lecture Notes in Computer Science 9285, Springer, Cham, 2015, pp. 157–172.
[14] A. Buzmakov, S. O. Kuznetsov, A. Napoli, Efficient Mining of Subsample-Stable Graph Patterns, in: 2017 IEEE International Conference on Data Mining (ICDM), IEEE, New Orleans, LA, 2017, pp. 757–762.
[15] R. Belohlavek, N. Kulkarni, V. Vychodil, A Novel Approach to Cell Formation, in: S. Ferré, S. Rudolph (Eds.), Formal Concept Analysis, Springer, Berlin, Heidelberg, 2009, pp. 210–223.
[16] M. J. Zaki, W. M. Jr., Data Mining and Analysis: Fundamental Concepts and Algorithms, Cambridge University Press, 2014.
[17] B. Ganter, S. O. Kuznetsov, Pattern Structures and Their Projections, in: H. S. Delugach, G. Stumme (Eds.), Proceedings of the 9th International Conference on Conceptual Structures, Lecture Notes in Computer Science 210, Springer, 2001, pp. 129–142.
[18] M. Kaytoue, S. O. Kuznetsov, A. Napoli, S. Duplessis, Mining gene expression data with pattern structures in formal concept analysis, Information Sciences 181 (2011) 1989–2001.
[19] A. Belfodil, S. O. Kuznetsov, M. Kaytoue, On Pattern Setups and Pattern Multistructures, International Journal of General Systems 49 (2020) 785–818.
[20] L. Szathmary, A. Napoli, P. Valtchev, Towards Rare Itemset Mining, in: Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), IEEE, 2007, pp. 305–312.
[21] S. O. Kuznetsov, N. Makhazhanov, M. Ushakov, On Neural Network Architecture Based on Concept Lattices, in: M. Kryszkiewicz, A. Appice, D. Ślęzak, H. Rybinski, A. Skowron, Z. W. Raś (Eds.), Foundations of Intelligent Systems, Springer International Publishing, Cham, 2017, pp. 653–663.
[22] A. Belfodil, S. O. Kuznetsov, C. Robardet, M. Kaytoue, Mining Convex Polygon Patterns with Formal Concept Analysis, in: C. Sierra (Ed.), Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI, ijcai.org, 2017, pp. 1425–1432.
[23] C. Demko, K. Bertet, J. Viaud, C. Faucher, D. Mondou, Description lattices of generalised convex hulls, International Journal of Approximate Reasoning 174 (2024) 109269.

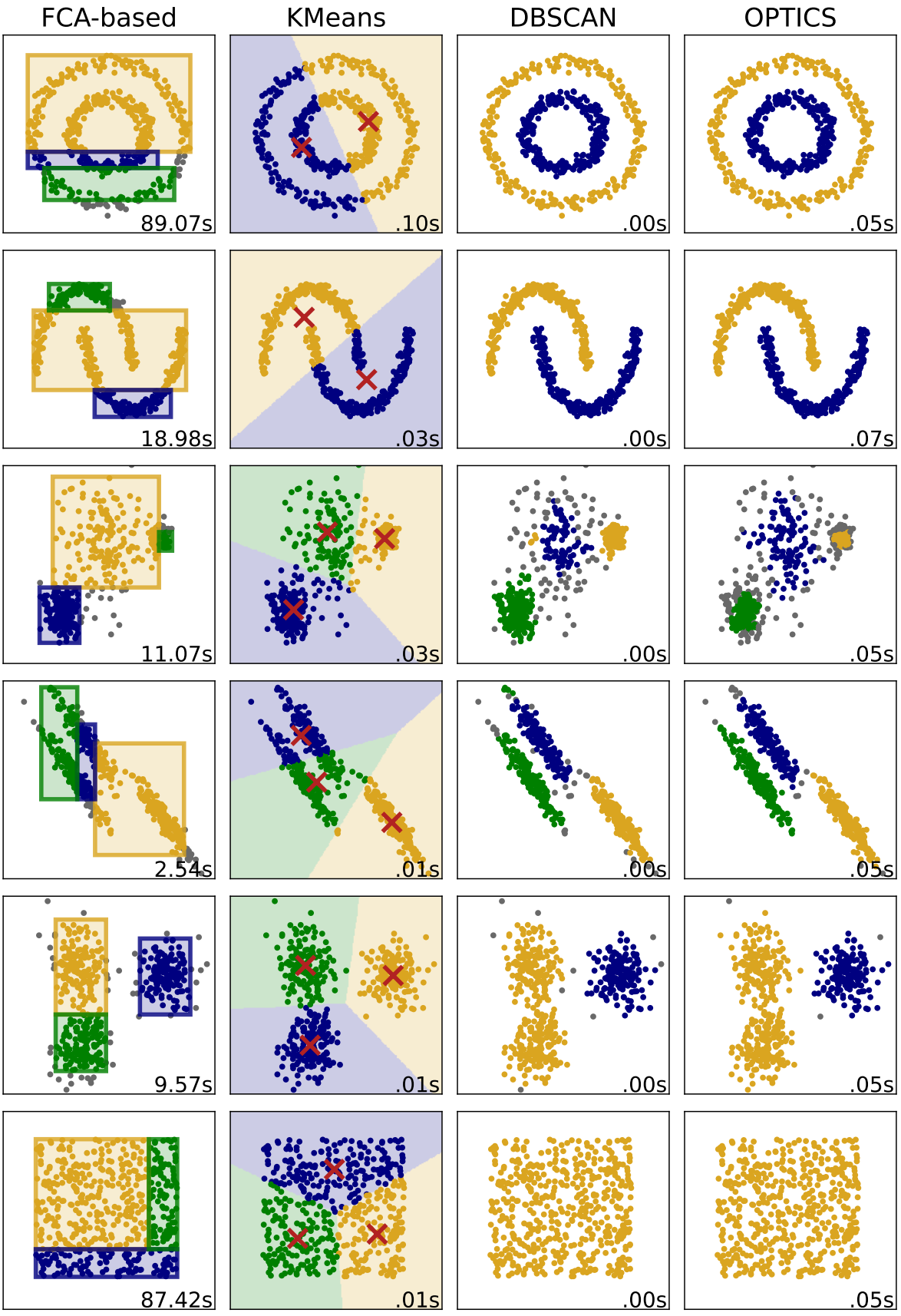

Figure 2: This visual comparison of the clusters produced by the different clustering approaches is inspired by the figure from Sci-Kit learn https://scikit-learn.org/stable/modules/clustering.html. The sets of dots having the same color correspond to clusters while sets of grey dots if any represent objects which are not belonging to any cluster.