# Explainable Document Classification via Pattern Structures

Sergei O. Kuznetsov(✉) and Eric George Parakal

National Research University Higher School of Economics, Pokrovsky Blvd, 11, Moscow 109028, Russia
{skuznetsov,eparakal}@hse.ru

**Abstract.** Inherently explainable Machine Learning (ML) models are able to provide explanations for their predictions by virtue of their construction. The explanations of a ML model are more comprehensible if they are expressed in terms of its input features. Our paper proposes an inherently explainable pipeline for document classification using pattern structures and Abstract Meaning Representation (AMR) graphs. The pipeline generates two kinds of explanations: intermediate and final ones, that justify its classifications. Intermediate explanations are represented as significant subgraphs found in the document graphs of test documents. Final explanations are the sentences of the test documents, that correspond to the significant subgraphs.

**Keywords:** Explainable Artificial Intelligence · Pattern Structures · Abstract Meaning Representation · Natural Language Processing

## 1 Introduction

The field of Explainable Artificial Intelligence (XAI) aims to explain the predictions of ML models so that their users can trust its predictions. While there exist a wide variety of XAI methods, an important distinction [1,6] is the one between *post-hoc explainability methods* and *inherently explainable models*. Post-hoc explainability methods explain a model's predictions after it has already been trained, whereas inherently explainable models incorporate explainability into their actual design.

Inherently explainable models have some distinct advantages over post-hoc explainability methods. It was demonstrated in [3] that the discriminative (predictive) power of a ML model is not correlated with its explainability, thus it cannot be guaranteed that a post-hoc explainability method can generate coherent explanations for a model with high discriminative power.

The author of [23] argued that inherently explainable models provide reliable explanations that reflect what the models actually learn, unlike the explanations provided by post-hoc explainability methods. [14] proved that simple models can have predictive performance that is nearly the same as more complex models,


S. Kovalev et al. (Eds.): IITI 2023, LNNS 776, pp. 423–434, 2023.
https://doi.org/10.1007/978-3-031-43789-2_39

thereby showing that the simplicity of inherently explainable models is not a detriment toward their predictive performance.

XAI methods often frame their explanations in terms of the input features of the models that they are explaining. [15] is an example of this idea that discusses different ways of perturbing various input features of Deep Neural Networks (DNNs) and observing the change in their output, in order to determine the importance of the input feature. The seminal paper of [17] also embodies this idea, by generating explanations that are comprised of the input features of the model represented as abstract, high-level, significant *concepts*.

Intuitively, perceptually similar examples of the input data having some semantic meaning, are termed as concepts. A concept is termed as being significant, if its presence is influential for the predictions of a ML model. The use of such concepts for explaining models belongs to a subfield of XAI methods called *concept-based explanations*.

In this paper, we propose an inherently explainable pipeline that classifies documents and generates explanations justifying the classifications, using pattern structures [12,16], see also a recent survey [10] and AMR graphs [2]. The vague, abstract notion of concepts is mathematically formalized via pattern structures, an applied lattice theory method; by describing concepts as groups of objects whose descriptions (called patterns) facilitate semilattice operations on the objects. AMR graphs are able to convert texts to graphs using a semantic representation language, so that the obtained graphs are able to ignore the syntactic idiosyncrasies of their respective texts.

Each document is converted into a document graph by using AMR graphs obtained from its sentences. Pattern structures use the training document graphs of each document class to build their respective conceptual hierarchies. The pipeline uses an aggregate rule that utilizes the conceptual hierarchy of each document class, to classify test document graphs as well as generate intermediate and final explanations.

The inherently explainable pipeline aims to have both good predictive performance as well as explainability, thereby disproving the stereotype about XAI which states that the predictive performance of a ML model must be sacrificed for its explainability and vice versa [9].

## 2 Related Work

While [17] introduced the idea of concept-based explanations, the original paper is an example of a post-hoc explainability method and consequently has some related flaws. The predominant one is that like all post-hoc explainability methods, it cannot guarantee a coherent explanation. This is because the method described in [17] cannot automatically identify any significant concepts, instead the user must supply the concepts themselves, in order to query their significance.

Papers such as [7] and [18] do not have this problem, as they present examples of inherently explainable concept-based models. The paper of [7] introduced the

process of concept whitening, wherein the latent space of a DNN is constrained so that its axes are aligned with some known interesting concepts. [18] proposed that an intermediate layer of a DNN is first trained to predict concepts, which are then trained to make the final classifications.

Applied lattice theory has been scarcely applied to the field of XAI. One of the earliest examples is the paper of [21], where Formal Concept Analysis (FCA) was used to construct explainable DNNs. The architecture of these DNNs was based on either antitone or monotone Galois connections, where every neuron is a formal concept.

In our paper [22] we proposed a similar inherently explainable document classification method, which we further elaborate here. Instead of using document graphs and pattern structures to build the conceptual hierarchies of each document class, we used binary attributes and standard FCA. The binary vectors representing the documents indicate the presence or absence of keywords, with the final explanations expressed as keyword clusters. Pattern structures are advantageous because they allow us to formalize many types of patterns, such as itemsets, interval tuples, graphs and sequence sets. However, since pattern structures are defined on meet-semilattices, pattern spaces that are only partially ordered cannot be modeled using pattern structures.

The paper of [4] explored this problem in detail and discussed the idea of pattern setups as a potential solution. Additionally, a new structure relying on multilattices, lying between pattern setups and pattern structures called pattern multistructures was proposed. The authors of [8] proposed the NextPriorityConcept algorithm, which facilitates the efficient mining of complex and heterogeneous data. The lattices obtained by the algorithm are smaller as the patterns of each concept are locally discovered. The algorithm can also merge patterns obtained from distinct space descriptions, which allows it to manage heterogeneous data in a generic and agnostic manner.

## 3 Basic Definitions

Here, we give basic definitions that are related to pattern structures from [12].

**Definition 3.1**: Let $G$ be some set of objects, then let $(D, \sqcap)$ be a meet-semilattice of potential object descriptions and let $\delta : G \rightarrow D$ be a mapping. Then $(G, \underline{D}, \delta)$, where $\underline{D} = (D, \sqcap)$; is called a **pattern structure**, provided that the set

$$\delta(G) := \{\delta(g) \mid g \in G\}$$

generates a complete subsemilattice $(D_\delta, \sqcap)$ of $(D, \sqcap)$, i.e., every subset $X$ of $\delta(G)$ has an infimum $\sqcap X$ in $(D, \sqcap)$ and $D_\delta$ is the set of these infima.

Instead of binary attributes, objects in a pattern structure have complex *descriptions* (patterns) with a *similarity operation* $\sqcap$ defined on them, so that $\delta(X) \sqcap \delta(Y)$ is a description of the similarity between the descriptions of $X$ and $Y$.

In this paper we use graph pattern structures, where descriptions of objects are graphs with labeled vertices and edges, and the similarity operation gives the set of maximal common subgraphs of the initial graphs.

**Definition 3.2**: If $(G, \underline{D}, \delta)$ is a pattern structure, the **derivation operator** is defined as

$$A^\diamond := \bigsqcap_{g \in A} \delta(g) \text{ for all } A \subseteq G$$

This is the set of maximal common subgraphs of graph descriptions, of objects in $A$. Correspondingly, for a pattern $d$, the derivation operator is defined as

$$d^\diamond := \{g \in G \mid d \sqsubseteq \delta(g)\} \text{ for all } d \in D$$

The elements of $D$ are called *patterns* and are ordered as

$$c \sqsubseteq d: \iff c \sqcap d = c$$

which is called the *subsumption* order, where $d$ *subsumes* $c$.

**Definition 3.3**: A *pattern concept* of $(G, \underline{D}, \delta)$ is a pair $(A, d)$ satisfying

$$A \subseteq G,\ d \in D,\ A^\diamond = d \text{ and } A = d^\diamond,$$

where $A$ is called the *pattern concept extent*, $d$ is called the *pattern concept intent* and the size of $A$ which is denoted by $|A|$, is called the *pattern concept support*.

The set of all pattern concepts forms the *pattern concept lattice*. In this work we refer to graph patterns as sets of graphs interchangeably, depending on the situation where the terms are used.

## 4 Methodology

The methodology of the inherently explainable pipeline consists of the following processes:

- Document to graph conversion: The DocToGraph algorithm converts a document into a document graph by first generating the AMR graphs of its individual sentences. The resulting AMR graphs are processed, refined and finally merged to obtain the document graph.
- gSOFIA algorithm [5] computes most stable (i.e., robust to noise in data) concepts for each document class, then supports of these concepts in all document classes are computed.
- Classification with an aggregate rule and intermediate explanation generation: A test document is classified by first determining its aggregate score for the graph pattern concept lattice, of each document class. The test document is classified as belonging to the class with the highest aggregate score. The intermediate explanations justifying the classification, are the significant subgraphs from the predicted class's graph pattern concept lattice, that are subsumed by the test document graph.

– Final explanation generation: The SubgraphToSent algorithm generates the final explanations from the intermediate ones, by mapping the significant subgraphs to their corresponding sentences in the test document. The mapping is done by finding those sentences of the test document, whose refined graphs can be minimally merged to obtain a graph, that is either equivalent to or subsumes the significant subgraphs.

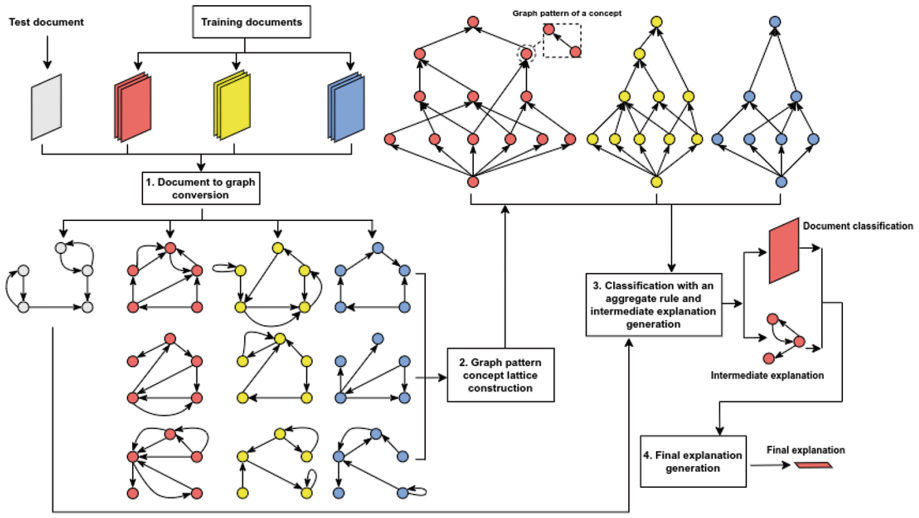


**Fig. 1.** Architecture of the inherently explainable pipeline.

We refer to Fig. 1 for the architecture of the inherently explainable pipeline, as well as a toy example that illustrates its functioning. Training documents belonging to three classes (red, yellow and blue) and a test document of an unknown class (silver) are shown. First, the DocToGraph algorithm converts each document to its respective graph. Second, the gSOFIA algorithm constructs the graph pattern concept lattice for each class. Third, the aggregate rule classifies the test document graph (as belonging to class red) using the constructed graph pattern concept lattices and also reveals its significant subgraph(s) (a class red one, belonging to the red graph pattern concept lattice), i.e., the intermediate explanations. Finally, the SubgraphToSent algorithm maps the significant subgraph(s) to the sentences in the test document, thereby obtaining the final explanations.

### 4.1 Document to Graph Conversion

The document to graph conversion process of the DocToGraph algorithm is partly inspired by the Communicative Discourse Tree (CDT) graph-based structure, proposed in [11].

Given a document $t$, the DocToGraph algorithm first generates the set of AMR graphs $\{a_i\}$ from the sentences of $t$. Each $a_i$ is generated independent of any other $a_i$, using the amrlib library[1] trained on the parse_xfm_bart_base[2] sentence to graph model. As stated in [2], AMR is heavily biased toward the English language and is not an Interlingua. Also, since each $a_i$ is generated independently, it is often the case that internal nodes with the same name are instances of different entities.

This problem can adversely affect the document classification performance of the aggregate rule. This is because the aggregate rule uses the subsumption relation that relies upon the commonality of node and edge labels, between the graph patterns of the graph pattern concepts and the test document graph.

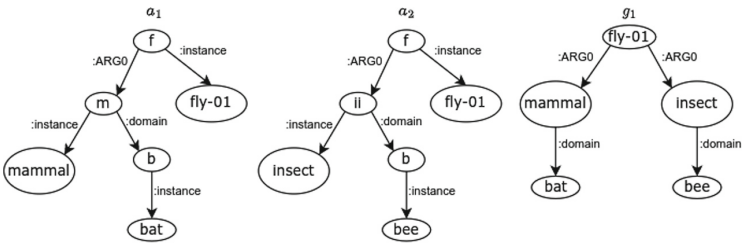


**Fig. 2.** A toy example of a document graph $g_1$ that is obtained after refining and merging the AMR graphs $a_1$ and $a_2$, of two sentences.

Figure 2 shows a toy example of a document having the following two sentences: "*A bat is a flying mammal. A bee is a flying insect.*" that is converted into a document graph $g_1$. We see that the AMR graph $a_1$ of the first sentence has an internal node $b$ that is an instance of the entity *bat*. However, in the AMR graph of the second sentence $a_2$, the internal node $b$ is an instance of the entity *bee*. This discrepancy is solved by replacing all of the internal nodes of an AMR graph, with their respective entities. Furthermore, the node labels of all AMR graphs are also converted to lowercase and lemmatized before merging the graphs (now called the set of refined graphs $\{r_i\}$), to obtain the document graph. This last step is done in order to maximize the document classification performance of the aggregate rule.

### 4.2 Classification with an Aggregate Rule and Intermediate Explanation Generation

The main idea of classification with an aggregate rule is as follows: for each class of documents, the $K$-most stable (w.r.t. to noise in composing data sample) pattern concepts are generated by means of gSOFIA. Every pattern intent of such

[1] https://github.com/bjascob/amrlib.
[2] https://github.com/bjascob/amrlib-models.

a concept, i.e., a graph or a set of graphs, is considered as a "weak" classification rule w.r.t. the corresponding class. All such rules are aggregated in an aggregate rule, taking into account its support in its respective class and its supports in other classes (which is penalized).

The use of aggregate rules for classification with pattern structures was proposed for "lazy" (or query-based) classification in [19,20]. The aggregate rule classifies a test document graph $g_{test}$ as belonging to the document class $c$ that has the highest aggregate score $A_{L_c}$. An aggregate score is computed for every graph pattern concept lattice $L_c$ of each class.

$$A_{L_c} = \frac{1}{|L_c|} \sum_{i \in L_c} \frac{support(i) \times |\delta(i)|}{penalty(i)} \big[\delta(i) \sqsubseteq \delta(g_{test})\big] \tag{1}$$

A graph pattern concept $i$ is penalized by the number of times its pattern intent, i.e., its graph pattern $\delta(i)$, is subsumed by training graphs $g_{train_{jk}}$ belonging to all classes, other than its own class $c$. The penalty of a concept is formally defined as $penalty(i) = \sum_{k \neq c}^{C} \sum_j \{1 \mid \delta(i) \sqsubseteq \delta(g_{train_{jk}})\}$, where $C$ is the set of all classes.

The aggregate rule is designed to give higher value to intents that are larger in size (shown by the weight $|\delta(i)|$) and are "contrast", i.e., they are well represented in their own class (shown by the weight $support(i)$), and are not well represented in other classes (shown by the $penalty(i)$). This allows the aggregate rule to suggest expressive intermediate explanations. Furthermore, the aggregate scores of each graph pattern concept lattice are normalized by the total number of concepts in that lattice, so as not to favor lattices with a larger number of concepts.

The contribution of a graph pattern concept $i$ toward classifying a test document as belonging to class $c$, is quantified by its share of the aggregate score $A_{L_c}$. The intermediate explanations for the classification of a test document are obtained from the graph patterns of the graph pattern concepts that belong to its highest scoring graph pattern concept $L_c$. The contribution of any graph pattern concept $i$ (from class $c$) for a test document graph $g_{test_{jk}}$, belonging to class $k$, is defined below.

$$contribution(i) = \frac{1}{|L_c|} \frac{support(i) \times |\delta(i)|}{penalty(i)} \big[\delta(i) \sqsubseteq \delta(g_{test_{jk}})\big] \tag{2}$$

A graph pattern (pattern intent) $g_{sub_{jc}}$ is called a significant subgraph, if its corresponding graph pattern concept has a high contribution to the highest aggregate score $A_{L_c}$, for a given test document.

### 4.3 Final Explanation Generation

The SubgraphToSent algorithm finds the minimally merged graph $M$, from a set of refined graphs $\{r_i\}$ of the sentences of a given test document $t_{test_{jk}}$, belonging to the document class $k$; that subsumes a given significant subgraph

$g_{sub_{jc}}$; from class $c$. This allows a mapping from $g_{sub_{jc}}$ to its relevant sentences in the test document $t_{test_{jk}}$, thereby acting as a set of final explanations.

Referring to Fig. 2, we can see that if any subgraph of $g_1$ is a significant subgraph, the minimal set of refined AMR graphs that can be merged to obtain the merged graph $M$ that will subsume the significant subgraph, will consist of the AMR graphs $a_1$ and $a_2$. Thus, the final explanations are the two sentences that correspond to the AMR graphs $a_1$ and $a_2$.

### 4.4 Complexity Analysis

The complexity analysis of the inherently explainable pipeline is performed by analyzing each of the individual sub-processes involved in its classification and explanation generation.

We begin with the conversion process from a document to an AMR graph. The complexity of generating the AMR graph of a sentence depends on the total number of words $w$ in that sentence, thus we get $O(w)$. For an AMR graph $a_i$, in order to replace all its internal nodes with their respective entities and further refine them, we have to traverse at most the total number of nodes in $a_i$. The total number of nodes in $a_i$ is denoted by $|a_i|$. Hence, the complexity of processing and refining an AMR graph is $O(|a_i|)$.

Since we use the gSOFIA algorithm [5] for generating the $K$-most stable pattern concepts for each document class $c$, and since gSOFIA has polynomial delay for checking subsumption $\sqsubseteq$, the complexity of computing these concepts can be given by $O(K \cdot p(\sqsubseteq))$, where $p(\sqsubseteq)$ denotes the complexity of computing $\sqsubseteq$.

The penalty and aggregate score functions defined in Sect. 4.2 both use the subsumption relation $\sqsubseteq$, in their computation. The penalty for a graph pattern concept $i$ belonging to class $c$, depends on the number of training documents $G_{train_{jk}}$ that belong to all classes other than $c$. Thus the complexity of the penalty function for any given $i$, is $O(p(\sqsubseteq) \cdot |G_{train_{jk}}|)$ where $c \neq k$. Similarly, the complexity of calculating the aggregate score $A_{L_c}$ depends on the number of graph pattern concepts in the graph pattern concept lattice $L_c$, so we have $O(p(\sqsubseteq) \cdot |L_c|)$.

## 5 Experiments and Results

The dataset [13] that we have analyzed (henceforth referred to as the BBC sports dataset) contains 737 documents (belonging to 5 classes) from the BBC Sport website that are related to sports news articles.

Due the small size of the datasets, a 90:10 training to test split ratio is used. As seen in Table 1 the aggregate rule provides excellent document classification performance for the BBC sports dataset. For the purpose of comparison, a baseline model; SVM is applied to the dataset after its texts have undergone TF-IDF vectorization (used here because of its simplicity). The result is also shown in

**Table 1.** Table describing the document classification performances.

| Model | Dataset | Macro-averaged $F_1$ score |
|---|---|---|
| Aggregate rule | BBC sports | 0.93 |
| SVM | BBC sports | 0.98 |

Table 1. Despite the fact that SVM outperforms the aggregate rule, it does not provide any explanations justifying its classifications.

We present the graph pattern concepts having the absolute highest contribution (as defined by Eq. 2) among all of the test documents for each class. Additionally, their contribution, their class prediction and the test document where they were found are shown in Table 2. These graph pattern concepts can then be said to be the most decisive in making the classification for their respective classes.

In the majority of the classes, the most decisive intermediate explanation is found in a test document of the class that it originally belongs to. However, even when this is not the case, the most decisive intermediate explanation is the reason for the misclassification. Thus, it acts as an explanation regardless of the correctness of the classification. The most decisive intermediate explanation having the highest contribution among all classes, is shown in Fig. 3.

**Table 2.** Table describing the most decisive intermediate explanation for each class of the BBC sports dataset.

| Class | Concept | Contribution | Prediction | Test document |
|---|---|---|---|---|
| Athletics | athletics_1401 | 0.01760 | Athletics | Athletics 8 |
| Cricket | football_101 | 0.03030 | Football | Cricket 11 |
| Football | football_296 | 0.14841 | Football | Football 2 |
| Rugby | football_260 | 0.05746 | Football | Rugby 10 |
| Tennis | tennis_1703 | 0.03781 | Tennis | Tennis 1 |

For the sake of conciseness, only the final explanation that is mapped from the most decisive intermediate explanation, shown in Fig. 3; belonging to the *football* document class, is written here. For the concept football_296, the relevant sentence from the football 2 test document is: "*Everton manager David Moyes will discipline striker James Beattie after all for his headbutt on Chelsea defender William Gallas.*" We can see that this sentence intuitively corresponds to the intermediate explanation, i.e., the significant subgraph from which it was derived.

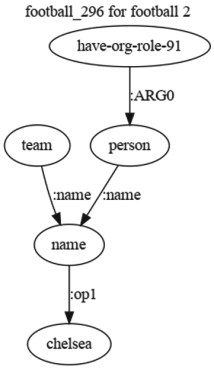


**Fig. 3.** A decisive intermediate explanation from the BBC sports dataset, belonging to the football document class.

## 6 Conclusion and Future Work

In this paper we have proposed an inherently explainable pipeline that can classify documents with excellent performance and provide reasonable explanations of performed classifications. The pipeline is able to achieve this using a novel combination of pattern structures and AMR graphs. Pattern structures are used for the classification and generating explanations, while AMR graphs are able to generate document graphs that can properly take advantage of the conceptual hierarchies obtained via pattern structures.

The results obtained in this paper can be extended to a recommender system that values transparency regarding its recommendations to its users. Future directions of research can deal with minimizing the complexity of the pipeline, as well as improving its classification performance. A promising path is to use the embeddings of the graph pattern concepts and their proximity to each other, for classification and generating the explanations. This will be computationally more efficient than using the aggregate rule and may provide better document classification performance, as well more meaningful explanations.

**Acknowledgements.** The work of Sergei O. Kuznetsov on this paper was supported by the Russian Science Foundation under grant 22-11-00323 and performed at HSE University, Moscow, Russia.

## References

1. Arrieta, A.B., et al.: Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. Inf. Fusion **58**, 82–115 (2020)
2. Banarescu, L., et al.: Abstract meaning representation for sembanking. In: LAW@ACL, pp. 178–186. The Association for Computer Linguistics (2013)
3. Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: quantifying interpretability of deep visual representations. In: CVPR, pp. 3319–3327. IEEE Computer Society (2017)

4. Belfodil, A., Kuznetsov, S.O., Kaytoue, M.: On pattern setups and pattern multi-structures. Int. J. Gen. Syst. **49**(8), 785–818 (2020)
5. Buzmakov, A., Kuznetsov, S.O., Napoli, A.: Efficient mining of subsample-stable graph patterns. In: ICDM, pp. 757–762. IEEE Computer Society (2017)
6. Carvalho, D.V., Pereira, E.M., Cardoso, J.S.: Machine learning interpretability: a survey on methods and metrics. Electronics **8**(8), 832 (2019)
7. Chen, Z., Bei, Y., Rudin, C.: Concept whitening for interpretable image recognition. Nat. Mach. Intell. **2**(12), 772–782 (2020)
8. Demko, C., Bertet, K., Faucher, C., Viaud, J.F., Kuznetsov, S.O.: Nextpriorityconcept: a new and generic algorithm computing concepts from complex and heterogeneous data. Theoret. Comput. Sci. **845**, 1–20 (2020)
9. Dosilovic, F.K., Brcic, M., Hlupic, N.: Explainable artificial intelligence: a survey. In: MIPRO, pp. 210–215. IEEE (2018)
10. Ferré, S., Huchard, M., Kaytoue, M., Kuznetsov, S.O., Napoli, A.: Formal concept analysis: from knowledge discovery to knowledge processing. In: Marquis, P., Papini, O., Prade, H. (eds.) A Guided Tour of Artificial Intelligence Research, pp. 411–445. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-06167-8_13
11. Galitsky, B.A., Ilvovsky, D.I., Kuznetsov, S.O.: Detecting logical argumentation in text via communicative discourse tree. J. Exp. Theor. Artif. Intell. **30**(5), 637–663 (2018)
12. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: Delugach, H.S., Stumme, G. (eds.) ICCS-ConceptStruct 2001. LNCS (LNAI), vol. 2120, pp. 129–142. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44583-8_10
13. Greene, D., Cunningham, P.: Practical solutions to the problem of diagonal dominance in kernel document clustering. In: ICML. ACM International Conference Proceeding Series, vol. 148, pp. 377–384. ACM (2006)
14. Hand, D.J.: Classifier technology and the illusion of progress. Stat. Sci. **21**(1), 1–14 (2006)
15. Ivanovs, M., Kadikis, R., Ozols, K.: Perturbation-based methods for explaining deep neural networks: a survey. Pattern Recognit. Lett. **150**, 228–234 (2021)
16. Kaytoue, M., Codocedo, V., Buzmakov, A., Baixeries, J., Kuznetsov, S.O., Napoli, A.: Pattern structures and concept lattices for data mining and knowledge processing. In: Bifet, A., May, M., Zadrozny, B., Gavalda, R., Pedreschi, D., Bonchi, F., Cardoso, J., Spiliopoulou, M. (eds.) ECML PKDD 2015. LNCS (LNAI), vol. 9286, pp. 227–231. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23461-8_19
17. Kim, B., et al.: Interpretability beyond feature attribution: quantitative testing with concept activation vectors (TCAV). In: ICML. Proceedings of Machine Learning Research, vol. 80, pp. 2673–2682. PMLR (2018)
18. Koh, P.W., et al.: Concept bottleneck models. In: ICML. Proceedings of Machine Learning Research, vol. 119, pp. 5338–5348. PMLR (2020)
19. Kuznetsov, S.O.: Fitting pattern structures to knowledge discovery in big data. In: Cellier, P., Distel, F., Ganter, B. (eds.) ICFCA 2013. LNCS (LNAI), vol. 7880, pp. 254–266. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38317-5_17
20. Kuznetsov, S.O.: Scalable knowledge discovery in complex data with pattern structures. In: Maji, P., Ghosh, A., Murty, M.N., Ghosh, K., Pal, S.K. (eds.) PReMI 2013. LNCS, vol. 8251, pp. 30–39. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-45062-4_3

21. Kuznetsov, S.O., Makhazhanov, N., Ushakov, M.: On neural network architecture based on concept lattices. In: Kryszkiewicz, M., Appice, A., Ślęzak, D., Rybinski, H., Skowron, A., Raś, Z.W. (eds.) ISMIS 2017. LNCS (LNAI), vol. 10352, pp. 653–663. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-60438-1_64
22. Parakal, E.G., Kuznetsov, S.O.: Intrinsically interpretable document classification via concept lattices. In: FCA4AI@IJCAI. CEUR Workshop Proceedings, vol. 3233, pp. 9–22. CEUR-WS.org (2022)
23. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. **1**(5), 206–215 (2019)