



Explainable Classification and Annotation through Relation Learning and Reasoning

Régis Pierrard

► To cite this version:

Régis Pierrard. Explainable Classification and Annotation through Relation Learning and Reasoning. Artificial Intelligence [cs.AI]. Université Paris-Saclay, 2020. English. NNT : 2020UPAST008 . tel-03115438

HAL Id: tel-03115438

<https://theses.hal.science/tel-03115438>

Submitted on 19 Jan 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Explainable Classification and Annotation through Relation Learning and Reasoning

Thèse de doctorat de l'Université Paris-Saclay

École doctorale n° 573, Interfaces : approches interdisciplinaires,
fondements, applications et innovations

Spécialité de doctorat : Informatique

Unité de recherche : Université Paris-Saclay, CentraleSupélec, Mathématiques et
Informatique pour la Complexité et les Systèmes, 91190, Gif-sur-Yvette, France.

Référent : CentraleSupélec

Thèse présentée et soutenue à Gif-sur-Yvette, le 15 septembre 2020, par

Régis PIERRARD

Composition du jury:

Hugues Talbot

Professeur des universités, CentraleSupélec

Freddy Lecue

Directeur de recherche, Thales/INRIA

Yann Chevalerey

Professeur des universités, Université Paris Dauphine

Isabelle Bloch

Professeur des universités, Télécom Paris

Céline Hudelot

Professeur des universités, CentraleSupélec

Jean-Philippe Poli

Ingénieur de recherche, Commissariat à l'énergie atomique et aux énergies alternatives (CEA)

Laurent Cabaret

Professeur agrégé, CentraleSupélec

Président

Rapporteur et Examineur

Rapporteur et Examineur

Examinatrice

Directrice

Encadrant

Invité

Acknowledgements

This thesis has been an extraordinary adventure and I would like to thank all the people that made its completion possible. First, I am thankful to all the members of the jury for the time they dedicated to evaluate my work and for their patience when we had to postpone the defense twice because of the exceptional sanitary circumstances.

I sincerely think the supervisors have a great impact on how well a thesis progresses and on the overall experience of the doctoral student. I was fortunate to be surrounded by people who were able to motivate and push me when it was needed and who helped me to achieve this work. Thus, I am very grateful to Jean-Philippe, my supervisor at CEA, who has always been available when I had a question or an issue and who gave me multiple suggestions to shape this thesis. I would also especially like to thank my thesis director, Céline Hudelot, for always supporting me and for being willing to meet and talk each time I needed it. Jean-Philippe and Céline always tried to put myself in the best position to carry out my thesis, which I appreciate very much.

A part of the work presented in this manuscript is the result of a collaboration with Laurent Cabaret, professor at CentraleSupélec. He helped me to deal with scientific constraints I had little knowledge about. I truly appreciated working with him and I would like to thank him for his availability and the time he spent reviewing a part of this work. I am also thankful to Isabelle Bloch for accepting to attend and review my mid-thesis defense and for her cooperation when I was working on the fuzzy dilation operator.

The warm welcome I received when I arrived in the LADIS laboratory and the people I met there also helped me to go through this whole experience. I am thankful to the midday eaters group and to all the people who played football every Wednesday after work. I am very pleased that this weekly football session has become a habit and I am looking forward to playing again. In particular, I would like to thank Sandra for the many breakfasts and talks we had together, for the hints she gave me and for all the activities she proposed, Vincent for the countless hours we spent in our office talking about every possible topic, Arnaud for his love of etymology and for the two great trips we did together in Spain and Brazil to attend conferences, and Ismaïl for the great technical and sport talks we had. I also want to thank Shivani, Andrei, Baptiste, Hung, Edwin, Étienne, Meritxell, Rafaël, Mickaël and Harry.

Finally, I would like to highlight the role of my family. I cannot thank my parents enough for their support, they have always encouraged me to give my best and given me everything I needed to follow the directions I chose. I am also grateful to my brother and my grandma for always being by my side and for keeping believing in myself.

Contents

Acknowledgements	iii
Introduction	1
I Explainability in AI	5
1 Explainable Artificial Intelligence	7
1.1 Towards a Definition of Explainability and Interpretability	8
1.2 Taxonomy	11
1.3 Related Works	13
1.4 Evaluating Explanations	32
1.5 Impact of Explanations on Users	36
1.6 Discussion	37
2 Proposed Approach	39
2.1 Which Explanation?	39
2.2 Which Model?	40
2.3 Which Features?	40
2.4 The Overall Approach	41
II Building an Explainable Model	45
3 Model Expressivity and Fuzzy Relations	49
3.1 Expressivity of a Model	49
3.2 Fuzzy Relations	51
3.3 Discussion	54
4 Learning Relevant Relations and Descriptors	55
4.1 Relation-based Descriptors	55
4.2 Frequent Itemset Mining	56
4.3 Fuzzy Frequent Itemset Mining	60
4.4 Fuzzy Close Algorithm	62
4.5 Discussion	68
5 Heuristics for Preventing Redundant Evaluations	71
5.1 Brute Force Evaluation of Relations	72
5.2 Online Pruning of Infrequent Relations	72
5.3 Knowledge-based Ordering of Relations	74
5.4 Discussion	84

6	Generating Rules or Constraints for Performing Explainable Classification or Annotation	87
6.1	Building Rules for Classification	87
6.2	Converting Relations into Constraints for Annotation	91
6.3	Generating Explanations from Rules and Constraints	95
6.4	Discussion	98
III	Application to Spatial and Temporal Reasoning	101
7	Spatial and Temporal Relations	105
7.1	Fuzzy Spatial Relations	105
7.2	Temporal Relations	114
7.3	Spatio-temporal Relations	115
7.4	Discussion	116
8	Fast Parallel Fuzzy Morphological Operators	117
8.1	Fuzzy Dilation	118
8.2	Related Algorithms	118
8.3	Vectorized Multithreaded Reverse Algorithm	123
8.4	Benchmark and Results	126
8.5	Discussion	130
8.6	Acknowledgements	130
9	Experiments on Images	131
9.1	Toy Dataset for Image Classification	131
9.2	Organ Annotation in Medical Images	140
9.3	Discussion	151
10	Application to Time Series Classification	153
10.1	Dataset	153
10.2	Vocabulary of Relations	155
10.3	Discussion	160
	Conclusion and Perspectives	163
A	Publications	169
A.1	International Peer-Reviewed Conferences	169
A.2	National Peer-Reviewed Conferences	169
B	Fuzzy Logic : Main Definitions	171
B.1	Fuzzy Set	171
B.2	Linguistic Variable	172
B.3	Fuzzy Relations	172
B.4	Fuzzy Operators	172
C	Closure Operator	175
C.1	Definitions	175
C.2	Proof	175
D	Fuzzy Close Algorithm: Experimental Results	179
D.1	Datasets	179
D.2	Results and Discussion	179

E Topological Sorting	181
E.1 Definitions	181
E.2 Example	181
E.3 Algorithms	182
F Additional Results	183
F.1 Constraints in the Organ Annotation Experiment	183
F.2 Time Series Classification: Linguistic Variables	184
G SIMD	185
G.1 Architecture	185
G.2 Instructions	186
H Résumé en français	187
Bibliography	189

List of Figures

1.1	Example of dog classification (Simonyan et al., 2013) which illustrates the difference between interpretation and explanation. On the left, the saliency map is an interpretation because it is relatively easy to understand given the input but it does not describe the logic that led to the classification. On the right, it is an example of explanation since the causes are clearly stated.	11
1.2	Example from (Laugel et al., 2019). Two classifiers have been trained on the Iris dataset, which contains three different classes. They both have an accuracy of 78% on the test set. In Figure 1.2A, the random forest makes some questionable divisions of the feature space due to its sensitivity to outliers. In Figure 1.2B, the SVM decisions far from training data do not seem reliable too. In those cases, post-hoc interpretability may produce plausible but misleading explanations.	13
1.3	An example of interpretable decision set from (Lakkaraju et al., 2016). Since it is a decision set, rules are independent from each other. The length of rules and their number per class is not high, which make the result understandable and the reasoning clear.	15
1.4	Example of decision tree for assessing what kind of lenses a person may wear. The reasoning leading to each class is clear and the shallowness of the tree makes the model easily interpretable. This example was taken from the course on decision trees given by Bhiksha Raj at Carnegie Mellon University.	16
1.5	Example where linear regression is used as an interpretable model (Poursabzi-Sangdeh et al., 2018). The goal is to predict the price of an apartment in a neighborhood of New York City given eight different features. The coefficients corresponding to each feature give users an idea of the role they play in the final result.	17
1.6	Example showing how kNN can be used to interpret a prediction (Kenny and Keane, 2019). The instance to classify is a “6”, but the system returns “0”. kNN enables to understand that the system has been misled by a few training instances that belong to the class “0” but look like “6”.	19
1.7	Overall architecture of the self-explaining neural network (Alvarez Melis and Jaakkola, 2018). There is a parametrizer (in orange) that generates relevance scores that correspond to β . Then, a encoder is used (in green) to learn higher level concepts from the raw features of the input space, which correspond to h . Finally, an aggregating layer enables to define g and an explanation is obtained as a set of couples coefficient/concept.	20

1.8	Example of explanation produced by a self-explaining network (Alvarez Melis and Jaakkola, 2018) on the MNIST dataset. Five concepts have been learnt and their corresponding coefficients are shown for two different inputs. For each concept, the most representative instances in the dataset are displayed.	20
1.9	Architecture of the network proposed by (Li et al., 2018). Inputs are projected into a latent space using an encoder so that they can be compared to the prototypes that were learnt. The decoder enables to visualize these prototypes.	21
1.10	Example from (Alonso and Bugarín, 2019). The goal is to predict if, following telemarketing, a consumer subscribes a term deposit. This is represented by the variable <i>Bank</i> that can take two values: “yes” or “no”. Here, the branch leading to the prediction is highlighted (in green) and an explanation in natural language is generated. We can see that the splitting criteria on the features <i>pdays</i> and <i>euribor3m</i> are characterized by the linguistic term “low”.	22
1.11	Optimization process that led to an image maximizing the activation of a single neuron (Olah et al., 2017). The initial image has been filled with random noise. At the end of the process, we can see the type of pattern that activates the neuron the most. In this particular case, the image maximizing the activation represents a regular pattern that seems to be a texture.	24
1.12	Example of a soft decision tree obtained by distillation of a teacher neural network on the MNIST dataset (Frosst and Hinton, 2017). The images at each node are the filters that were learnt to define the probability distributions used for splitting subsets of instances. The most likely classifications at each node and at each leaf are annotated in the tree.	25
1.13	A set of prototypes and a set of criticisms that were learnt on the MNIST dataset (Kim et al., 2016). We can see that prototypes look more like handwritten digits that a human would expect than criticisms.	25
1.14	This plot represents the Individual Conditional Expectancy of a prediction with respect to feature F_1 (Goldstein et al., 2015). We can see that there is a parabolic relationship between the estimator \hat{y} and F_1 . The dots correspond to the actual value of F_1 for each instance. The yellow line is the Partial Dependence Plot, which is the average of the ICE over all instances.	27
1.15	Architecture of the model presented in (Kim et al., 2018). In ①, the user provides a set of examples representing a concept (here “striped”) and a set of random examples. In ②, the training instances corresponding to the class under study (here zebras) are provided. In ③, f_l is the projection of an input instance into the latent space corresponding to layer l , and $h_{l,k}$ is the classifier. In ④, the concept activation vector \mathbf{v}_C^l is learnt by training a linear classifier to distinguish between the activations produced by the concept’s examples and the ones produced by other examples. Finally, in ⑤, the conceptual sensitivity is computed by deriving $h_{l,k}$ with respect to $f_l(\mathbf{x})$ in the direction of \mathbf{v}_C^l	28

1.16	The model presented in (Hendricks et al., 2016) extracts visual features and predicts a class label. Then, sentence generation is conditioned by both the visual features and the class label to get a discriminative and descriptive explanation.	29
1.17	Illustration of the principles of LIME and Anchors.	31
1.18	Architecture of the model presented in (Lécué and Pommellet, 2019). An object detection model is trained and applied on an image. In parallel, contextual information is extracted from knowledge graphs for each type of object in the dataset. Depending on the consistency between this contextual information and the predictions of the other objects in the image, the confidence of the output of the model increases or not.	33
1.19	Example from (Hendricks et al., 2016) that presents the difference between image descriptions, which are not necessarily class relevant, and class definitions, which are not necessarily image relevant. A visual explanation should rely on class discriminative features that are also present in the image under study.	35
2.1	Illustration of the approach proposed in this thesis for performing image annotation.	42
3.1	Figure representing the linguistic variable ("duration", $[0; 100]$, $\{F_{\text{short}}, F_{\text{long}}\}$). The membership functions of the two fuzzy sets F_{short} and F_{long} are represented respectively in blue and red. The core of F_{short} is $[0; 25]$ and the core of F_{long} is $[75; 100]$	53
3.2	Two figures where the relation <i>disk to the left of square</i> cannot be expressed the same way. Fuzzy logic provides tools to characterize differently these two situations while relying on the same fuzzy relation <i>to the left of</i>	54
5.1	Curves representing the evolution of $B = \frac{n(S-1)}{k} + 1$ with the number of assessed examples k for a total number of examples $n = 30$. For the sake of clarity, only the positive values of B are displayed (if B is negative, no support can be discarded because it is always positive or null).	74
5.2	Example of labeled directed graph. The vertices are relations and the directed edges between vertices are labeled according to the links between relations. For each edge, its corresponding labels are to its right.	80
5.3	Example of directed acyclic graph that we get using Algorithm 3 on the knowledge graph displayed on Figure 5.2.	82
7.1	Pictures representing the 8 core relations of RCC8: DC (Disconnected), EC (Externally Connected), PO (Partially Overlaps), EQ (Equals), TPP (Tangential Partial Part), NTPP (Non-Tangential Partial Part), TPPi (Tangential Partial Part inverse) and NTPPi (Non-Tangential Partial Part inverse) (Randell et al., 1992).	106
7.2	An example of fuzzy dilation. Figure 7.2B represents the fuzzy landscape associated to the fuzzy dialtion of the red object by the structuring element displayed in Figure 7.2A.	109
7.3	Example of distance relations "near" and "far".	110
7.4	Example of a fuzzy landscape representing the relation "between" the two red objects (Cinbis and Aksoy, 2007).	111

7.5	Examples from (Vanegas, 2011).	111
7.6	Examples of enlacement between two objects (Clément et al., 2017). . .	113
7.7	Axes of symmetry found by the method described in (Colliot, 2003) . .	114
7.8	Examples of fuzzy temporal scopes (Poli et al., 2018).	116
8.1	Fuzzy dilation $D_{SE}(F)$ of the reference object F by the structuring element SE . The spatial relation represented here is “close to”. The intensity of each pixel of $D_v(\mu)$ represents in which extent it verifies the relation. The image of the structuring element is four times as big as the two others because it is needed for performing the dilation.	118
8.2	For each iteration of the fuzzy dilation, $D_{SE}(F)(u)$ with $u \in \mathcal{U}$, the structuring element SE is looped over the input image F . As shown on these two figures, SE needs to be bigger than F to cover it completely.	119
8.3	Comparisons of fuzzy landscapes generated with both the naive algorithm and Bloch’s method.	122
8.4	Comparison of the naive and reverse algorithms for computing a fuzzy dilation. The input image has 384 rows. While the naive approach computes the dilation pixel by pixel, the reverse one only computes the contributions of the non-zero pixels in the reference object. On row 33, the first non-zero pixels of the reference object are barely perceptible on the figure and have been surrounded by a red ellipse. Once the reference object has been completely looped over (row 170), the reverse algorithm finishes and returns the same result as the naive approach after row 384.	124
8.5	Contribution to the fuzzy dilation of one active pixel (in red) from the reference object (on the left). For the sake of this illustration, only 4 threads and 128 columns are represented. PR is the multi-threaded version of the <i>reverse</i> algorithm where each thread is responsible for a strip of rows of the fuzzy dilation $D_v(\mu)$. With PR_{128} , PR_{256} and PR_{512} , for each thread, columns of $D_v(\mu)$ are distributed using AVX, AVX2 and AVX512 respectively.	126
8.6	Artificial dataset samples: a) Rectangle, b) Disk, c) Ellipse, d) 256×256 crisp square with 256 active pixels, e) 256×256 crisp square with 4096 active pixels, f) 256×256 crisp square with 65536 active pixels, g) 256×256 fuzzy square with 256 active pixels, h) 256×256 fuzzy square with 4096 active pixels, i) 256×256 fuzzy square with 65536 active pixels, j) 512×512 crisp square with 65536 active pixels, k) 1024×1024 crisp square with 65536 active pixels	127
8.7	Execution time distributions for the natural image dataset.	130
9.1	Examples from each class of the dataset used in the example of explained classification in Section 9.1	132
9.2	Example of how a specific relation is computed in an instance. Here, the goal is to compute the relation <i>ellipse to the right of disk</i> . Given an instance (Figure 9.2A), the disk, which is the reference object in the relation to evaluate, is extracted (Figure 9.2B). The fuzzy landscape <i>to the right of disk</i> can then be computed (Figure 9.2C). Finally, as explained in Chapter 7, the relation can be evaluated using a fuzzy pattern matching approach.	133

9.3	Example of signature of a fuzzy ellipse generated using (Chanussot et al., 2005). For each alpha cut of the ellipse, its edge is extracted and the distance of each point of this edge to the centroid of the ellipse is computed. Thus, we get a shape signature for each alpha cut. At the end, the final signature is obtained by averaging all the signatures we got at the previous step. Here, we can see that there are two maxima and two minima of same values. Also, these extrema are uniformly spread over the edge. This signature is typical of an ellipse.	134
9.4	Graph representing the logical links between the relations in the vocabulary. There are three subgraphs with only one type of edge: e , which represents $\mathcal{R}_1 \Rightarrow \mathcal{R}_2$	135
9.5	Plot displaying the performance of the model with respect to the value of the minimum support. We reach an accuracy of 100% between 0.5 and 0.65. The red dashed line represents the average number of relations in rules before they were pruned from relations common to other classes. The red plain line is its counterpart for rules that were pruned from common relations.	136
9.6	Example of explanation for an instance from class 1. Relations in bold were pruned because they are shared by rules from other classes. . . .	138
9.7	Example of explanation for an instance from class 2. This is one of the borderline example we generated. Relations in bold were pruned because they are shared by rules from other classes.	138
9.8	Example of explanation for an instance from class 3. Relations in bold were pruned because they are shared by rules from other classes. . . .	139
9.9	Example of explanation for an instance from class 4. This is one of the borderline example we generated. Relations in bold were pruned because they are shared by rules from other classes.	139
9.10	Examples of the four types of scans in the dataset.	140
9.11	In this experiment, we consider the 9 colored organs in this figure. . . .	141
9.12	Comparison of the fuzzy landscapes corresponding to the relation <i>to the left of the liver</i> (figure 9.12A on page 142) and the relation <i>completely to the left of the liver</i> (figure 9.12B on page 142).	142
9.13	Examples of symmetry measure.	142
9.14	Graph representing the logical links between the relations handled by the model in this experiment.	142
9.15	Example of explained annotations.	145
9.16	Answers to the first four assertions.	149
9.17	Answers to the last four assertions.	150
10.1	Examples from the time series dataset we worked on.	154
10.2	Three examples from class 4 that do not present the same properties at first sight. Those are the three types of patterns that we encounter in the instances of class 4.	154
10.3	Example of a time series from class 1. The 8 signals have been smoothed and a segmentation has been performed. The vertical black line represents the border between the first segment, corresponding to transient state, and the second segment, corresponding to steady state.	155
10.4	Graph representing the logical links between relations for the time series classification experiment.	156
10.5	Example of explanation for an instance from class 1.	157
10.6	Example of explanation for an instance from class 2.	157

10.7	Example of explanation for an instance from class 3.	158
10.8	Example of explanation for an instance from class 5.	158
10.9	Confusion matrix displaying the performance of our model in the time series classification experiment.	159
10.10	Given an input image, we can get a convenient segmentation of the entities of interest by applying hierarchical clustering on an overseg- mentation.	166
D.1	Plots showing the number of database passes relatively to the mini- mum support threshold for the three datasets.	180
E.1	Example of a directed acyclic graph for topological sorting.	182
F.1	Fuzzy sets corresponding to the linguistic variables <i>Increases</i> and <i>De- creases</i>	184
F.2	Fuzzy sets corresponding to the linguistic variable <i>Varies</i>	184
F.3	Fuzzy sets corresponding to the linguistic variable <i>Distance</i>	184
G.1	Figure representing how SIMD instructions enable to parallelize com- putations compared to a single instruction operating on single data (Intel, 2011).	185
G.2	Figure showing how different data types fit in a 128-bit register (upper part of the figure) and in a 256-bit register (lower part) (Intel, 2011). . .	186

List of Tables

1.1	Table representing the taxonomy we propose for XAI methods. The approaches classified in this table are described in Section 1.3. There is no example of method that relies on both transparency and model-agnosticism since, by definition, model-agnostic methods do not rely on any model.	33
1.2	Criteria for evaluating explanations (Baaj and Poli, 2019). They are split into three categories: <i>natural language</i> , <i>human-computer interaction</i> and <i>content and form</i> . The first category aims at assessing the correctness of the language used in explanations. In the second category, criteria enable to evaluate what the explanation conveys when it is transmitted from the system to the user. The third category is dedicated to assessing the content and the form of the explanation.	35
4.1	Example of database for the <i>basket market problem</i> . We call this database $\mathcal{D}_{\text{market}}$	57
4.2	Representation of the formal context associated to $\mathcal{D}_{\text{market}}$ (cf. Table 4.1). Zeros are not displayed for a better visualization of the formal context.	57
4.3	A fuzzy database $\mathcal{D}_{\text{fuzzy}}$ represented as a fuzzy formal context.	61
4.4	The fuzzy database $\mathcal{D}_{\text{fuzzy}}$	66
4.5	FCC_1 on the left and FC_1 on the right. $\{\text{🍌}\}$ is pruned from FCC_1 to FC_1 because it is not frequent.	69
4.6	FCC_2 on the left and FC_2 on the right.	69
4.7	FC	69
4.8	Deriving frequent itemsets. Bold lines refer to derived itemsets. From left to right: L_3 , L_2 and L_1	69
5.1	The fuzzy database $\mathcal{D}_{\text{fuzzy}}$. For a minimum support greater than or equal to 0.5, we can know that $\{\text{🍌}\}$ cannot be frequent after having processed the first four transactions. However, for $\{\text{🍉}\}$, we need its evaluation in t_5 to assess whether it is frequent or not.	72
5.2	This table shows the four types of implications between relations that the approach can process. Those implications enable to propagate the result of one relation to another. \mathcal{R}_1 and \mathcal{R}_2 are two p -ary fuzzy relations defined on a space A . e is a tuple of entities defined on A^p	78
5.3	Recap of the different kinds of link we consider between relations and their notation in the graph representation. The third column specifies how the corresponding edge is represented in a graph. \mathcal{R}_1 and \mathcal{R}_2 are two p -ary fuzzy relations defined on a space A	79
6.1	The set FC of frequent closed itemsets that we got in the example of Chapter 4 on page 55.	90

7.1	Definition of topological relations in the original RCC (Randell et al., 1992) and the fuzzy RCC for regions u and v in a universe \mathcal{U} (Schockaert et al., 2009). t is a left-continuous t-norm and $\overset{t}{\rightarrow}$ the residual implicator corresponding to t	106
8.1	Execution time in ms for one fuzzy landscape computation with variable image sizes and a 4096-pixel centered fuzzy square. Bold font indicates the best result for each case.	128
8.2	Execution time in ms for one fuzzy landscape computation with variable image sizes and a 65536-pixel centered fuzzy square. Bold font indicates the best result for each case.	128
8.3	Speedups with the <i>reverse</i> algorithm as reference for variable image and reference object sizes. Bold font indicates the best result for each case.	128
8.4	Execution time in ms for one fuzzy landscape computation with variable number of active cores and reference object sizes.	129
8.5	Acceleration ratio with <i>Bloch's</i> algorithm as reference with variable image and reference object sizes.	130
9.1	Table representing several results for each class of organ when performing a nested cross-validation with 5 folds in the outer loop. First, the minimum support associated to the learning of the frequent subsets of relations is given for each class. In the second column, the average number of constraints that we get for each class of organ at the end of the learning is displayed. Finally, we show the average confidence that we got for each annotation during the testing (accuracy of 100%).	146
9.2	Table presenting the number of prevented evaluations using our second heuristic (cf. section 5.3 on page 74). Following the topological sort that we got in section 9.2.2 on page 141, the results for each logical link are shown. Overall, this strategy enables to prevent 1636 evaluations, which represents 7.6% of the total number of evaluations.	147
10.1	The second column shows the values of minimum support we found for each class after tuning. The third column shows the average length of rules before the relations that are shared by rules from other classes are removed. The last column displays the average length of rules after they have been pruned.	159

Introduction

Context

Artificial Intelligence (AI) has reached many layers of society and has become more and more popular. It is naturally part of many current and developing technologies, such as smartphones and autonomous vehicles for instance. It has also turned into a hot topic of discussion among politicians, social scientists or in the media. This keen interest in AI is the consequence of the great strides the field has made over the last decade, in particular with the success of Deep Learning (DL). Deep neural models have enabled to dramatically improve the performance of automatic systems in many different tasks, such as object recognition (Russakovsky et al., 2015) or also high-level reasoning tasks like strategic games (Silver et al., 2016). While a few articles claim to reach superhuman performance (Russakovsky et al., 2015; Brinker et al., 2019), their main purpose is to interact with humans and help their decision-making. On the other hand, this great coverage about the recent successes of AI also contributes to put forth its current weaknesses.

This is not the first time AI has raised so much enthusiasm since the field was born in 1956. The motivation behind AI has often been linked to humans and their activity, hence interactions between both parts. AI applications aim at either replicating a task performed by humans, like automatic quality control in the industry for example, or at aiding humans perform more complex assignments, such as diagnostic-aid systems in medicine. The latter is an example of critical application since the decision of the AI may have an impact on the well-being of the patient. In such high-stake applications, interactions between AIs and humans are even more important. As a consequence, the cost of making a bad decision is very high (Rudin, 2019) and so great levels of safety, robustness and reliability are required. However, the most performing approaches do usually not satisfy these requirements and may even be vulnerable to attacks (Szegedy et al., 2013; Nguyen et al., 2015; Moosavi-Dezfooli et al., 2016). Thus, they are not suited to such applications.

In that context, several areas of research have been investigated in the last few years to build *trustable* AIs. Those are *privacy*, *fairness* and *explainability*. Privacy aims at preserving individuals against potential attacks and weaknesses of AI systems that could reveal information about them. Fairness is also a growing topic in AI because datasets that are used to build state-of-the-art models are likely to contain biases. Therefore, the model will replicate these biases, which will be detrimental to its performance and its acceptance. Explainability aspires to provide the rationale behind a model and its decisions. This enables to understand why a model returns a given output.

In particular, explainability has recently received special attention and the name *eXplainable Artificial Intelligence* (XAI) has been introduced by DARPA (Gunning, 2016) and is now commonly accepted to refer to this field. XAI is composed of methods that enable to build transparent models, to get more insight into what a black-box model is doing or to evaluate explanations. This field is actually not new since explainability was already a concern at the time of the first rule-based expert

systems in the 1970's. Because of the weaknesses of current AIs, recent laws have been proposed to regulate their use (European Council, 2016; Assemblée Nationale, 2019). Coupled with a need for gaining acceptance in the society, it has prompted a resurgence of interest in XAI. This has led the research community to propose new approaches.

The present work aims at integrating the platform *ExpressIF* developed within CEA. In this platform, an interpretable and fuzzy-logic-based AI is implemented. Interpretability being one of its core features, several operators were created or integrated so that the knowledge in the model is directly expressible in natural language and thus more understandable to humans. In this thesis, we would like to learn from such a large vocabulary of operators for building an interpretable model whose decisions can be explained.

Problematic and Goals

Our objective is to propose a new approach that enables to build models able to both classify/annotate 1D or 2D signals and provide an explanation. This thesis does not have a targeted application and our aim is to propose a generic approach that could be applied to various application cases. Nevertheless, in our work, we are concerned by *high-stake applications* whose impact on humans is important and thus a *human-understandable* explanation of the decision is mandatory.

For high-stake applications, it has been claimed that black-box models are not appropriate because getting a faithful and detailed enough explanation is tricky (Rudin, 2019). Indeed, in most cases, it consists in only attributing importance scores to features (Ribeiro et al., 2016; Lundberg and Lee, 2017) that are not necessarily interpretable. In such critical applications, *transparent models*, i.e. models whose trace and reasoning can be conveniently tracked, are better suited for producing reliable explanations. As a consequence, in our approach, we focus on transparent models, such as decision rules, and thus we do not resort to more opaque ones like deep neural networks.

Understanding a signal such as an image or a time series relies on *understanding the relations* between the entities in those signals (Biederman, 1981; Geurts, 2001). Thus, we assume in this work that these relations should be a core component of our approach. Therefore, we would like our model to be able to express and handle such relations to classify/annotate instances as well as to generate explanations. Moreover, in order to bring knowledge about potentially relevant relations, we are interested in a *human in the loop* approach that would enable to set beforehand a vocabulary.

In the context of critical applications, it may be difficult to gather (labeled) data and so we may have to deal with small datasets. For our approach to be generic, it should be able to learn relevant relations (from the vocabulary set beforehand) on few data. Thus, one of our goal is to develop a learning strategy that is suited to any dataset sizes. This problematic is also linked to the choice of the transparent model.

Explanation evaluation is of paramount importance for assessing the quality of the explanations returned by the model. This is a major issue in the field of XAI, even though a few directions have been proposed (Doshi-Velez and Kim, 2017; Hendricks et al., 2018; Baaj and Poli, 2019). In this thesis, we do not focus on proposing a new explanation evaluation approach but we will adapt *human-grounded* evaluation (Doshi-Velez and Kim, 2017; Baaj and Poli, 2019) to our needs.

Contributions and Outline

This thesis is organized into three parts as follows.

The first part of this document gives a broad view of the field of XAI. Since there is no reference definition of explainability, we first specify one in order to set a frame for the explanations that our model will produce. Following a taxonomy of XAI methods that we proposed, we present an extensive glimpse of state-of-the-art methods. This enables us to spot the advantages and weaknesses of these methods and to propose a new approach that can learn relevant relations in data to classify or annotate instances with explanations. Besides, we describe the current strategies for evaluating explanations.

In the second part, we present the theoretical foundations of our approach. In Chapter 3, we specify the notion of expressivity of our model and present the tools we rely on to build an expressive model and to express interpretable relations. Chapter 4 tackles our **first contribution**, which is the learning strategy we propose to build a model. It is inspired by frequent itemset mining and enables to learn from few data. It is also suited to learn on correlated data such as the instances of a given class. It extracts class descriptors as frequent sets of relations that will be the basis for classification/annotation and explanations. Our **second contribution** is presented in Chapter 5 and consists in two heuristics that aim at making the evaluation of relations faster. The first heuristic relies on the properties of the learning algorithm to discard relations that are bound to be infrequent. The second heuristic is based on the logical links between relations, such as implications or dependencies. It results in an adequate order of evaluations of relations for propagating information and preventing redundant computations. In Chapter 6, we describe how we build rules or constraints from the relevant relations. We also present how we generate explanations in natural language.

The third and last part presents applications to spatial and temporal relations for dealing with images and time series. Chapter 7 is an overview of fuzzy spatial, temporal and spatio-temporal relations. In Chapter 8, we present our **third contribution** which consists in a fast, parallel and SIMD-based implementation of the fuzzy dilation operator to make the evaluation of some relations faster. The last two chapters, Chapter 9 and Chapter 10, present the experiments we carried out in order to test our approach. Chapter 9 tackles image applications and in particular an example of explainable organ annotation in medical images. Chapter 10 presents an application to time series classification on toxic chemicals.

Finally, we will conclude this thesis with an evaluation of our work and we will present some envisioned prospects.

Part I

Explainability in AI

Chapter 1

Explainable Artificial Intelligence

Artificial Intelligence (AI) has seen a great renewal in the last decade with the emergence of *Deep Learning* (DL). It enables to build models that are able to very efficiently classify images, recognize objects or translate from one language to another. It has been argued that it can even tops human capabilities on several specific tasks (Russakovsky et al., 2015; Silver et al., 2016; Haenssle et al., 2018). As a consequence, deep neural networks have been deployed in many applications, including use cases where the decisions they produce have a direct impact on human well-being. Indeed, the stakes are not the same for an autonomous vehicle as for performing music recognition for instance.

For high-stake applications of AI, performance is not the only criterion to optimize (Doshi-Velez and Kim, 2017). Such applications may require a relative understanding of the logic performed by the AI. In that case, the end-user would like to get a response to the question “Why?” (Miller, 2017). *eXplainable Artificial Intelligence* (XAI) aims at providing methods that help to make AIs able to answer this question.

The designation *explainable artificial intelligence* was first used in (Gunning, 2016). The author states that XAI applications should fulfill two objectives:

1. building models that are more explainable than those obtained with usual *Machine Learning* (ML) techniques while keeping a high level of performance,
2. being understandable, trustable and manageable to humans.

The first objective is actually not necessarily compatible with optimizing performances. It is usually admitted that there exists a trade-off between the performance of a model and its interpretability (James et al., 2013), although this claim may not be legitimate (Rudin, 2019). A simpler model may not be as accurate as a more complex one, but it should be more interpretable.

The second objective is more related to the applications and recipients of XAI. It should be used to make AI applications trustworthy (such as in high-stake applications), easier to improve (by spotting biases for example) or to assess the accountability of an autonomous system when an incident happens for instance.

The rise of the need for explainability in AI has also prompted governments to introduce new regulations. The most famous one is the *General Data Protection Regulation* (GDPR) that was introduced by the European Union in 2016 and that has been enforced since 2018 (European Council, 2016). In particular, articles 13 and 14 state that the data subject shall be provided “*meaningful information about the logic, as well as the significance and the envisaged consequences of such processing for the data subject*”. This has led researchers to wonder whether it introduces a *right to explanation* (Goodman and Flaxman, 2017) or more simply a *right to information* (Wachter et al., 2017a). Although it is never mentioned explicitly, all the necessary conditions for a right to explanation seem to be present in the GDPR (Selbst and Powles, 2017). In

1974, the US government introduced the *Equal Credit Opportunity Act* (United States House of Representatives, 1974) to prevent discriminations against credit applicants. It already mentioned that, when a creditor rejects a request for credit, it must give the applicant a statement of the reasons for this decision. For instance, such reasons can be “*The proportion of your revolving balances to total balances is too high*” or “*You recently opened a new account*”. With the extensive use of credit scores in the USA, such as the FICO score (FICO, 1989), this amounts to explaining why an applicant was given a low score by a model evaluating his/her/its trustworthiness. In 2018, FICO launched an Explainable Machine Learning Challenge (FICO, 2018) to improve the explainability of the ML models the company uses for generating credit scores. More recently, in October 2019, the French National Assembly approved a draft bioethic law (Assemblée Nationale, 2019) that introduces, in article 11, a *human guarantee*: when a medical decision-aiding model is used, its results must be revealed to the patient by a healthcare professional that will inform him/her about how the model is used and how it works. It also states that the optimization process of the model parameters requires the intervention of a healthcare professional, who may not have any knowledge in AI.

While regulations do exist, their requirements regarding the explainability of AI models are technically vague. This is the reflect of the current state of the field of XAI. Indeed, there is no agreement at the moment on reference definitions of explanations and interpretations, nor on how to evaluate and present them. Those are ongoing issues that involve multiple different fields, such as AI, cognitive sciences or law.

Besides, in order to assess, certify or compare models, methods for evaluating explanations are needed. This is a tricky topic because two different models may produce explanations in different modalities or two different people may expect different explanations depending on their knowledge.

In this chapter, we first deal with the definitions of explainability/interpretability and explanation/interpretation. We will present the definitions that have been proposed and will specify our position regarding this topic. Then, in Section 1.2, we propose a taxonomy of XAI methods, depending on what they explain and how they are applied. We will rely on this to present in Section 1.3 a review of the different methods that have been proposed by the community. In Section 1.4, we tackle the issue of evaluating explanations. Finally, in Section 1.5, we present a brief overview of the impact of explanations on users.

1.1 Towards a Definition of Explainability and Interpretability

The field of XAI involves people with different backgrounds such as ML, symbolic AI, Cognitive Science and Law. Several definitions of explainability and interpretability have been proposed in the literature with influences from the fields of expertise of their authors. As a consequence, there are currently no consensual definitions of these core notions. (Lipton, 2018) suggests that any claim about interpretability should rely on a specific definition of this notion. That is why we focus in this section on rendering an overview of the definitions that have been proposed, before specifying the definitions we relied on in this thesis.

1.1.1 Definitions from the Oxford Dictionary

The word *explain* comes from the Latin *explanare*, which means to make clear or, more literally, to make flatten. The Oxford Dictionary of English defines the verb *explain* as to make an idea or situation clear to someone by describing it in more detail or revealing relevant facts. The word *interpret* comes from the Latin *interpretari*, which is derived from the noun *interpretes* that means an agent or translator between two parties. The Oxford Dictionary of English defines it as to explain the meaning of information or actions. This definition involves *to explain*, which shows that the two notions are very close to each other. The etymology of *interpret* is also related to the notion of intermediary between two parties, which is not the case for *explain*.

1.1.2 Definitions from Artificial Intelligence

In the context of XAI, explainability and interpretability are often linked to transparency, reliability and trust. That means that they are not directly tied to the performance of a system but rather to its usage. Also, they are often used interchangeably.

The explainability of an AI is its ability to generate an explanation. Therefore, we first have to define what an explanation is. There are two main types of explanation. The most obvious one is the category of causal explanations. Halpern and Pearl (Halpern and Pearl, 2001) define it as a fact that, if found to be true, would constitute an actual cause of a specific event. Here, an explanation is the answer to a “why” question. Whereas a non-causal explanation is the answer to the question “what happened” (Ginet, 2016). But in XAI the goal is to return an explanation representative of the logic that led to a decision, which is exactly what the GDPR states. As a consequence, the kind of explanation that is expected in XAI is a causal one.

In the fields of AI and ML, several definitions have already been proposed. Based on how explanations are defined in social sciences, Miller (Miller, 2017) defines an explanation as the conjunction of three elements:

- a process of abductive inference to determine an explanation for a given event in which the causes are identified;
- a product, which is the result of the latter point;
- a process of transferring knowledge between the explainer and the explainee.

So an explanation is a product of a process of abductive inference and it is communicated from the system to the explainee. Gilpin et al. (Gilpin et al., 2018) proposed another definition. They think an explanation is defined by two features:

- *interpretability*, whose goal is to describe how the system works in terms understandable to humans;
- *completeness*, whose goal is to describe how the system works in an accurate way.

According to this definition, the main purpose is to find explanations that are both interpretable and complete. There is usually a trade-off between interpretability and completeness. This trade-off depends on how much knowledge related to the application the explainee has. However, this definition relies on the concept of interpretability that sometimes has its own definitions.

In 1953, Tarski gave a first definition of *interpretability* in the context of first-order logics (Tarski et al., 1953). For two theories¹ T_1 and T_2 , T_2 is said to be *interpretable* in T_1 if we can extend T_1 such that this extension is also an extension of T_2 . More informally, that means that there must be a common ground between the facts that an AI (T_2) and humans (T_1) can express for the AI to be interpretable to humans. Modern definitions also convey this idea. Doshi-Velez and Kim (Doshi-Velez and Kim, 2017) define interpretability as the “*ability to explain or to present in understandable terms to a human*”. They acknowledge that finding a formal definition of explanation is complicated and maybe impossible. (Biran and Cotton, 2017) define interpretability as the degree to which an observer can understand the cause of a decision. Based on this definition of interpretability, Miller (Miller, 2017) states that an explanation is one mode of obtaining understanding. Thus, for him, interpretability and explainability are equivalent. (Ribeiro et al., 2016) states that an interpretable model “*can be readily presented to the user with visual or textual artifacts*”. Older definitions from subfields of AI like fuzzy logic link the interpretability of a system to its complexity. For example, a rule-based expert system with 1000 rules was not considered interpretable because of the high number of rules, even though it might be able to explain a particular decision (Jin, 2000).

The conclusion is that there is no clear consensus around one definition. Some authors even use the two concepts interchangeably. Still, what emerges from this study is that interpretability aims at presenting elements that contributed to a decision in an understandable way, while explainability intends to logically and accurately render the decision-making process. Furthermore, we noticed that most definitions mention humans since they are often the main target of XAI methods. Thus, making understandable to humans also includes constraints on the medium (human language, vocabulary related to the knowledge of the explainee) and the complexity of the output. Thus, we propose the two following definitions:

Definition 1: *interpretability*

Interpretability is the ability to extract the elements that contributed to a decision in a way that is understandable for humans.

Definition 2: *explainability*

Explainability is the ability to describe to humans how a system works in a way that is accurate and that logically renders the reasoning of the system.

In the following of this work, we will talk about explainability and interpretability according to these two definitions.

To illustrate the difference between these two notions, Figure 1.1 displays an example of classification. The input image represents a dog. The output “It is a dog” states that this image belongs to the class “dog”. One interpretation is given as a saliency map (Simonyan et al., 2013). This shows the impact that each pixel of the input had on the classification. This is insightful when it is coupled with the original image, but the saliency map alone is not necessarily understandable to humans. Also, it does not give any information about the reasoning that the model performed to return this output. One example of explanation is also given. The causes are clearly stated: it has fur, it has a tail and it has a dog’s nose. Those three facts are consistent with the knowledge in the model, which leads to the output

¹A theory that is formalized within the first-order predicate logic is a set of axioms that are composed of variables and constants.

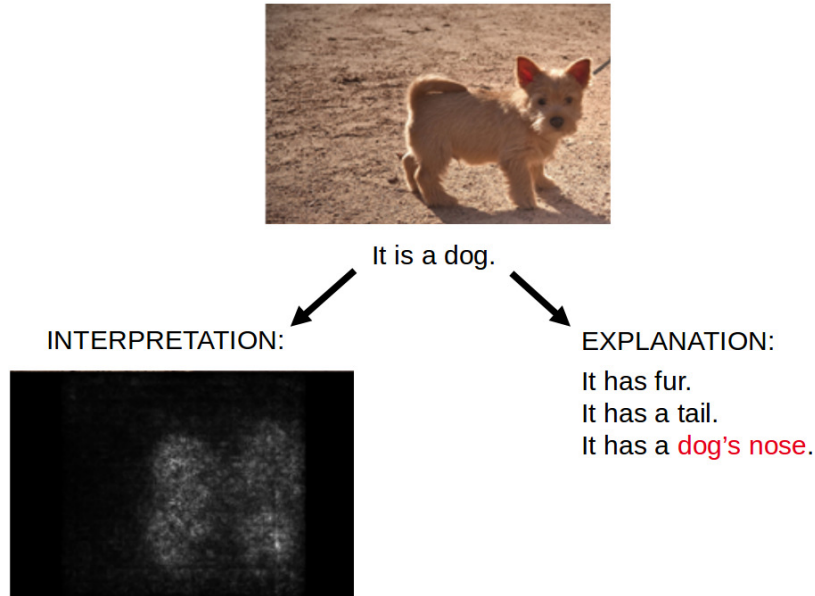


FIGURE 1.1: Example of dog classification (Simonyan et al., 2013) which illustrates the difference between interpretation and explanation. On the left, the saliency map is an interpretation because it is relatively easy to understand given the input but it does not describe the logic that led to the classification. On the right, it is an example of explanation since the causes are clearly stated.

(which is not necessarily true). Thus, the end user knows the logic involved in this decision and where it comes from.

Here, the interpretation has been performed by returning visual clues about the input. It could also be clues about the parameters of the model or extracting decisive features with a feature extraction method. However, for the explanation, a medium of communication that can express the reasoning has to be used. For most end users, natural language is the most obvious and trustworthy medium to perform it. As we see in Section 1.4, the mode of an explanation has an impact on the way humans assess it.

The concept of *justifiability* is also sometimes used. It differs from explainability and interpretability because it only aims at legitimating a decision without being necessarily close to how the system works (Miller, 2017).

1.2 Taxonomy

Depending on the objective and on the nature of the model, different methods from the XAI literature may apply. In this section, we focus on providing a taxonomy of XAI methods that specifies in which case one method is appropriate. The taxonomy we propose is inspired by (Molnar, 2019) and relies on the three following criteria:

- resorting to a *transparent* model to have straightforward explanations from it, or applying a *post-hoc* interpretability method on a usually more opaque model to get insight into what it is doing,
- explaining the *global* behaviour of the model or, at a *local* scale, explaining a specific decision on a single instance,
- is the method *model-specific* or *model-agnostic*?

1.2.1 Transparent Model or Black-Box Model with Post-hoc Interpretability

We make a first distinction between models that are inherently transparent and models that are seen as black boxes.

Lipton introduced three kinds of *transparency* (Lipton, 2018):

- *Simulatability*, which is transparency at the scale of the whole model. That means that one can have a grasp of the complete model at once. More than on the type of model, it mainly depends on its size. For example, a very deep decision tree is probably not as interpretable as a forest composed of few shallow trees. This is actually dependent on human ability to understand a more or less large model, which is subjective.
- *Decomposability*, which is the transparency of the model at a modular level. This consists in interpreting or explaining what a part of the model does. This can be one or a few rules in an expert system, or the parameters of a linear regression.
- *Algorithmic transparency* at the level of the learning algorithm. While the algorithm itself is fully transparent and understandable, the properties of the learning process may not. For instance, the optimization process that enables to train a deep neural network is not well understood and is still an active research topic.

In this taxonomy, we talk about transparency as Lipton's simulatability. Thus, what we call transparent models enable to immediately get an understanding of what they do. However, they may not perform as well as black-box models.

Post-hoc interpretability consists in generating interpretations after a model has been completely set. While this type of methods does usually not enable to obtain a global understanding of the model, they are able to extract information that may be valuable. Their two main advantages is that they can be applied to black-box models and they do not sacrifice predictive performance. That is why they are often used with deep neural networks. The counterpart is that it cannot catch the logic of the whole model and it can be misleading if it is not applied properly. This may happen when the optimization process fulfills a subjective demand (Lipton, 2018) or when it takes into account artifacts resulting from the learning process (Laugel et al., 2019). Laugel et al. give the example of counterfactual explanations, which are generated by finding the smallest perturbation to the input that changes the prediction (a more detailed description of what counterfactual explanations are is given in Section 1.3.2.2). It is thus highly sensitive to artifacts that may result from the learning process. Figure 1.2 shows an example where a few areas of the feature space would lead to misleading counterfactual explanations.

1.2.2 Global or Local

A *global* approach aims at describing the global behaviour of a model. On the other hand, the purpose of a *local* approach is to explain only a single instance.

For example, feature importance in random forests (Ho, 1995) (cf. Section 1.3.2.1) is global because it only considers the global behaviour of the forest. On the other hand, for a given instance, computing the Shapley values (Shapley, 1953) (cf. Section 1.3.2.2) for each feature is a local approach since it depends on the feature values of the instance.

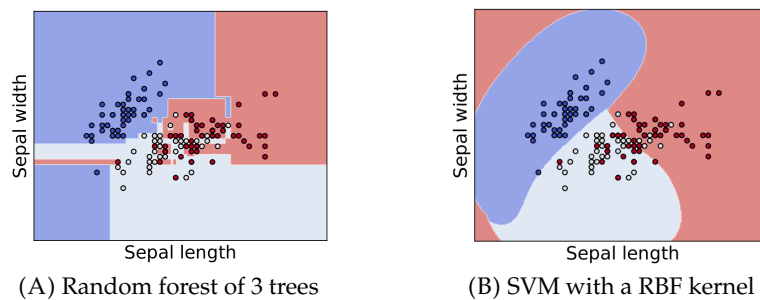


FIGURE 1.2: Example from (Laugel et al., 2019). Two classifiers have been trained on the Iris dataset, which contains three different classes. They both have an accuracy of 78% on the test set. In Figure 1.2A, the random forest makes some questionable divisions of the feature space due to its sensitivity to outliers. In Figure 1.2B, the SVM decisions far from training data do not seem reliable too. In those cases, post-hoc interpretability may produce plausible but misleading explanations.

1.2.3 Model-specific or Model-agnostic

A *model-specific* method is, by definition, specific to a model or a family of models. The scope of use of such methods is thus restricted.

A *model-agnostic* method is able to deal with any model. Such methods are always applied *post-hoc* since they do not rely on the intrinsic properties of the models they try to explain.

Let us consider again the example about feature importance and Shapley values. Feature importance is specific to decision trees and ensemble methods involving them, such as random forests or XGBoost (Chen and Guestrin, 2016). Thus, this is a model-specific method. Shapley values are model-agnostic since they only need to compute the output of a model regardless of its internal working.

1.3 Related Works

This section aims at reviewing the state-of-the-art methods in XAI. While there has been recently a surge of interest for explaining the decisions returned by AIs, this is not a new problem. The first expert system was Dendral (Feigenbaum et al., 1970; Lindsay et al., 1993). It was released in 1965 for describing the molecular structure of unknown organic chemical compounds. It was not able to explain its outputs but it could provide a trace of the program reasoning steps. The first intelligent systems that were able to provide an explanation were developed in the 1970's. Among them, MYCIN was one of the first rule-based expert systems (Shortliffe and Buchanan, 1975). It was an automated consultation system for improved antimicrobial selection. It returned an explanation as a subset of rules that led to the final result. Then, decision trees and bayesian networks enabled to get interpretable models. The architecture of such trees and networks is interpretability-friendly since they make the reasoning clearer. Later, the emergence of deep learning made Artificial Intelligence improve in terms of performance. However, explainability issues rose again because of the difficulty to explain what is happening inside deep neural networks (Marcus, 2018).

According to the taxonomy we proposed, we can distinguish models or methods that are either transparent or performing post-hoc interpretability, either global

or local, and either model-specific or model-agnostic. We will specify where each model/method is classified in this scheme.

In the following, many methods rely on machine learning concepts. A model is trained on a training set and is then tested on a test set. Thus, we introduce the following notations. Let $\mathcal{D} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\} \subseteq \mathcal{X} \times \mathcal{Y}$ be a training set of n instances where \mathbf{x}_i is the i -th feature vector and y_i its corresponding label (classification) or target value (regression). The set \mathcal{F} of m features corresponding to the instances in \mathcal{D} is $\{F_1, \dots, F_m\}$. The tuple (\mathbf{x}, y) will refer to an instance from the test set.

1.3.1 Transparent Models

As stated in the previous section, in this thesis, we define *transparency* as the ability to understand a whole model at once. It is model-dependent and so no model-agnostic method is presented in this subsection (which explains the cross in Table 1.1 on page 33).

1.3.1.1 Global

As we wrote in the introduction of this section, **expert systems** were already able to provide elements of explanation in the 1970's (Shortliffe and Buchanan, 1975). In particular, rule-based expert systems consist in an inference engine and a knowledge base. The knowledge base is composed of facts and *if-then* rules while the inference engine applies these rules to known facts to infer new facts. Thus, for each output, such systems are able to provide the trace of the rules which led to the result. That is why their reasoning is transparent. However, since the knowledge base of an expert system is usually big, the trace may not provide an intelligible explanation.

Decision rules have also been used to build other types of models and learning algorithms have been proposed to avoid generating handcrafted rules (Agrawal et al., 1993; De Raedt and Thon, 2011). With the resurgence of interest in XAI, several rule-based methods have been proposed in the last few years (Letham et al., 2015; Lakkaraju et al., 2016; Malioutov et al., 2017). In particular, (Lakkaraju et al., 2016) propose to learn a set of rules, called *decision set*, that can be used independently from each other. The independence between rules is interesting for generating explanations because it ensures to get a short trace. So the length of the explanations will only depend on the length of the rules. In their work, a rule is a tuple (I, c) where I is an *itemset*, which is a conjunction of predicates of the form (feature, operator, value) (e.g. $(F_j, \leq, 1)$), and $c \in \mathcal{Y}$ is a class label. An instance \mathbf{x} is assigned label c if it satisfies I . A decision set is also characterized by four metrics:

- its *size* (number of rules),
- the *lengths* of its rules (the number of predicates in a rule),
- the *covers* of its rules (the set of instances that satisfies a rule),
- the *overlap* of two rules (the set of instances that satisfies both rules).

Coupled with the accuracy of each rule, these metrics enable to evaluate the whole decision set. Decision sets are built in two steps: (1) a frequent itemset mining (Agrawal et al., 1993) algorithm is applied to extract a set of itemsets, (2) a learning

objective is optimized such that it favours smaller decision sets and shorter and non-overlapping rules. Thus, since rules are independent from each other, this should enable to provide explanations that are concise enough for humans. An example of explainable decision set learnt with Lakkaraju's method is displayed in Figure 1.3.

If Respiratory-Illness=Yes and Smoker=Yes and Age \geq 50 then Lung Cancer

If Risk-LungCancer=Yes and Blood-Pressure \geq 0.3 then Lung Cancer

If Risk-Depression=Yes and Past-Depression=Yes then Depression

If BMI \geq 0.3 and Insurance=None and Blood-Pressure \geq 0.2 then Depression

If Smoker=Yes and BMI \geq 0.2 and Age \geq 60 then Diabetes

If Risk-Diabetes=Yes and BMI \geq 0.4 and Prob-Infections \geq 0.2 then Diabetes

If Doctor-Visits \geq 0.4 and Childhood-Obesity=Yes then Diabetes

FIGURE 1.3: An example of interpretable decision set from (Lakkaraju et al., 2016). Since it is a decision set, rules are independent from each other. The length of rules and their number per class is not high, which make the result understandable and the reasoning clear.

Decision trees are another family of models that make their reasoning appear clearly. A decision tree is a tree structure where each node corresponds to a splitting test on a specific feature and where leaves correspond to the intended result. They can be used for classification and regression (Breiman, 1984), but we focus here on classification trees. Going from the root to a leaf, one gets a conjunction of splitting tests that enable to classify an instance. The most critical point is the construction of these splitting tests. The main methods for building decision trees are:

- *ID3* (Quinlan, 1986), which proposes to start from a root node that represents the whole dataset. Then, at each iteration, the feature corresponding to the greatest information gain is selected and used to split the dataset into as many subsets as possible values for this feature. The selected feature is then removed from the possible splitting features. The information gain obtained by splitting the feature $F_i \in \mathcal{F}$ in the dataset $\mathcal{D}' \subseteq \mathcal{D}$ is the difference between the entropy in \mathcal{D}' and the sum of the entropies in the subsets of \mathcal{D}' obtained by this splitting. The entropy $H(\mathcal{D}')$ in the dataset \mathcal{D}' is defined as follows:

$$H(\mathcal{D}') = - \sum_{c \in \{y_1, \dots, y_n\}} p_{\mathcal{D}'}(c) \log(p_{\mathcal{D}'}(c)) \quad (1.1)$$

with $p_{\mathcal{D}'}(c)$ the proportion of instances in \mathcal{D}' that belong to class c . The information gain $I_F(\mathcal{D}')$ can then be defined:

$$I_F(\mathcal{D}') = H(\mathcal{D}') - \sum_{\mathcal{D}'' \in \text{split}(\mathcal{D}', F)} p_{\mathcal{D}'}(\mathcal{D}'') H(\mathcal{D}'') \quad (1.2)$$

with $\text{split}(\mathcal{D}', F_i)$ the set of subsets that are obtained by splitting \mathcal{D}' with respect to feature F_i .

The drawback of this method is that it is not suited to continuous features or discrete features with many different values.

- C4.5 (Quinlan, 1993) is an extension of ID3 that can handle continuous values. For a continuous feature, all its values in \mathcal{D} are sorted in ascending order. Then, looping on these sorted list of values, different threshold are generated for splitting feature values. At the end, the selected threshold is the one that enables to maximize information gain.
- CART (Breiman, 1984) differs from C4.5 on a few aspects (Wu et al., 2008), including in particular:
 - Splits in CART are always binary (two outcomes) while C4.5 splits a dataset into *at least* two subsets at each node,
 - CART's splitting criterion is based on the *Gini index* while C4.5 relies on entropy. The Gini index is defined as follows:

$$G(\mathcal{D}') = \sum_{c \in \{y_1, \dots, y_n\}} p_{\mathcal{D}'}(c)(1 - p_{\mathcal{D}'}(c)). \quad (1.3)$$

For each split, the goal is to find the split that minimizes the Gini index.

The rules that we get at the leaves of a decision tree can be easily extracted to obtain the reasoning of the model. However, due to a high number of nodes, a deep tree may not lead to explainable rules and to a clear view of the overall model. In Figure 1.4, an example of explainable decision tree is given.

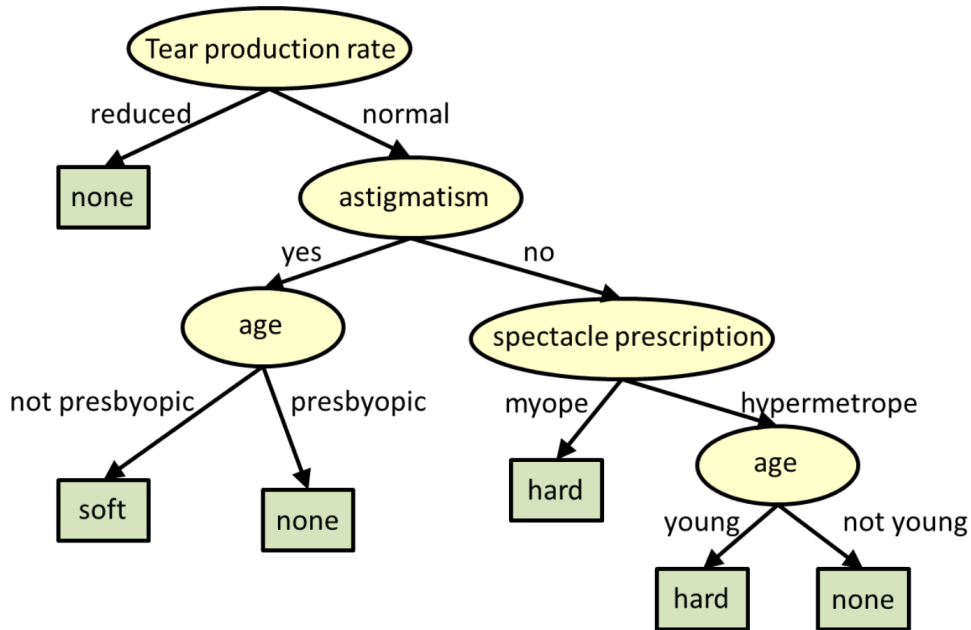


FIGURE 1.4: Example of decision tree for assessing what kind of lenses a person may wear. The reasoning leading to each class is clear and the shallowness of the tree makes the model easily interpretable. This example was taken from the course on decision trees given by Bhiksha Raj at Carnegie Mellon University.

Besides, tree-based ensemble methods, which consist in combining several decision trees to produce the intended results, are not considered as globally transparent models since they require a strategy for aggregating the outputs of the trees.

Another type of model that is transparent is **linear regression**. For a continuous output variable, it assumes that its relationships with input variables (features) are linear. Thus, for an instance $\mathbf{x}_i = (x_{i,1}, x_{i,2}, \dots, x_{i,m})$, an estimation of the output variable is

$$\hat{y}_i = \sum_{j=1}^m \beta_j x_{i,j} + \varepsilon, (\beta_1, \dots, \beta_m, \varepsilon) \in \mathbb{R}^{m+1} \quad (1.4)$$

where ε is an error term and β_j is the coefficient associated to $x_{i,j}$. This kind of model is often fitted using a least square approach that may be penalized by an L_1 -norm penalty (known as *lasso* (Tibshirani, 1996)) or a L_2 -norm penalty (known as *ridge regression*). Figure 1.5 shows an example of linear regression used in the context of XAI: users are given the values of coefficients $\beta_1, \dots, \beta_m, \varepsilon$ so that they can understand the role of each feature in the final result. While the reasoning of such a model may not be as intuitive as *if-then* rules in decision rules and trees, it is still understandable as long as the number m of features is small enough. In particular, it is convenient for extracting the most important features.

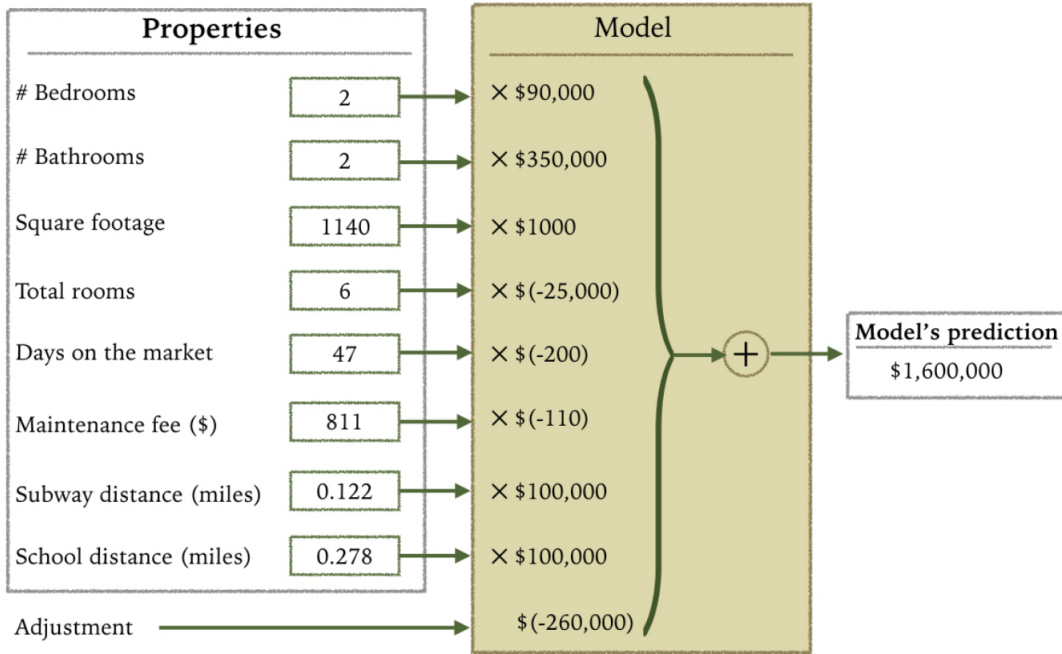


FIGURE 1.5: Example where linear regression is used as an interpretable model (Poursabzi-Sangdeh et al., 2018). The goal is to predict the price of an apartment in a neighborhood of New York City given eight different features. The coefficients corresponding to each feature give users an idea of the role they play in the final result.

Generalized Linear Models (GLM) (Nelder and Wedderburn, 1972) have been proposed to generalize linear regression. In GLM, the expected value of a label $y \in \mathcal{Y}$ is a function of the linear predictor presented above:

$$\forall y \in \mathcal{Y}, E(y | \mathbf{x}_i) = f\left(\sum_{j=1}^m \beta_j x_{i,j}\right) \quad (1.5)$$

with f the *link* function and $\sum_{j=1}^m \beta_j x_{i,j}$ the linear predictor (keeping the same notations as in Equation (1.4)). For example, we get the original linear regression if f is the

identity function, and a logistic regression if f is the sigmoid function. However, the link function may make the model less interpretable than a linear regression.

It is possible to generalize this concept further. While GLM assume linear relationships between the features and the target variable, **Generalized Additive Models** (GAM) (Hastie and Tibshirani, 1986) assume non-linear relationships. The linear predictor is now an additive predictor, such that:

$$\forall y \in \mathcal{Y}, E(y \mid \mathbf{x}_i) = f\left(\sum_{j=1}^m g_j(x_{i,j})\right) + \varepsilon \quad (1.6)$$

with f the link function, g_1, \dots, g_m non-linear functions and $\varepsilon \in \mathbb{R}$ the error term. While this type of methods is sometimes said to be interpretable, the combination of the link function and the non-linearities make it much less clear than the other transparent models we presented in this section.

1.3.1.2 Local

Naive Bayes classifiers are a well-known family of models in the ML community. They are a particular kind of bayesian networks where features are assumed to be independent on each other conditionally to the class. This assumption enables to build simple networks that do not contain any latent variable. According to Bayes' theorem, we have for an instance $\mathbf{x}_i = (x_{i,1}, x_{i,2}, \dots, x_{i,m})$:

$$p(c \mid \mathbf{x}_i) = \frac{p(c, x_{i,1}, \dots, x_{i,m})}{p(\mathbf{x})}, \forall c \in \{y_1, \dots, y_n\} \quad (1.7)$$

Relying on the independence assumption and the Bayes' theorem, one can show that

$$p(c, x_{i,1}, \dots, x_{i,m}) = p(c) \prod_{j=1}^m p(x_{i,j} \mid c), \forall c \in \{y_1, \dots, y_n\} \quad (1.8)$$

Thus, it is possible to directly assessed the impact of each feature on the classification of an instance. Furthermore, this can be represented as a graphical representation, which makes this approach more interpretable.

Another type of approaches consists in returning the closest concepts (Alvarez Melis and Jaakkola, 2018), instances (Kenny and Keane, 2019) or prototypes (Li et al., 2018) to the instance under consideration. This is especially suited for images since concepts or instances can be visualized.

The simplest way to achieve that is to use the **k-nearest neighbors** (kNN) algorithm. Its principle is to assign to an instance the class that is the most represented among its k closest neighbors. So the performance of the model strongly depends on the value of k and on the distance metric used to assess the closeness between instances. The interpretability of this approach relies on the instances (from the neighborhood) that were used to produce the result. So, ultimately, it depends on how interpretable a single instance is, which may not be suited to many applications. Figure 1.6 displays an example where it provides valuable insights. One image representing a "6" has been classified as a "0". Looking at its nearest neighbours, one can see that they all represent a "0" but look like a "6". In this example, it helps the user to understand that a few training instances are misleading.

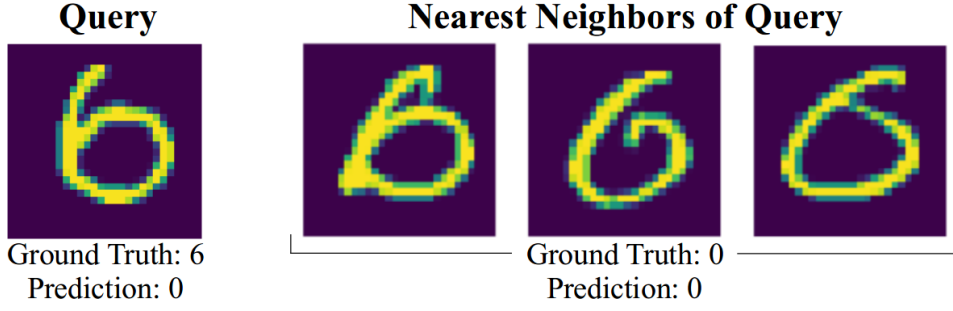


FIGURE 1.6: Example showing how kNN can be used to interpret a prediction (Kenny and Keane, 2019). The instance to classify is a “6”, but the system returns “0”. kNN enables to understand that the system has been misled by a few training instances that belong to the class “0” but look like “6”.

The approach proposed by (Alvarez Melis and Jaakkola, 2018) is similar to GAM and GML and aims at obtaining the concepts that contributed to the final decision. Their model relies on three characteristics:

- Unlike linear regression, the coefficients β_j depend on the input \mathbf{x} . We write them $\beta_j(\mathbf{x})$. More precisely, the coefficients are generated by a model that returns a vector $\beta(\mathbf{x})$ of m coefficients (m being the number of features) for any instance \mathbf{x} . It must respect the constraint that, for two close instances \mathbf{x} and \mathbf{x}' , $\beta(\mathbf{x})$ and $\beta(\mathbf{x}')$ should not differ significantly. This constraint ensures local interpretability.
- Since raw features may not always be the most interpretable units to provide a feedback to humans, the model should manipulate higher level features that are more suited to human understanding. Thus, a mapping $h: \mathcal{X} \mapsto \mathcal{X}' \subseteq \mathbb{R}^k$ is learnt on the training set. k should be relatively small so that the explanations are concise enough for humans. The generalized model is now, for an instance (\mathbf{x}_i, y_i) :

$$\hat{y}_i = \sum_{j=1}^k \beta_j(\mathbf{x}_i) h_j(\mathbf{x}_i) \quad (1.9)$$

- In Equation (1.9), the elements $\beta_j(\mathbf{x}_i) h_j(\mathbf{x}_i)$ are summed to produce the final result. The authors propose to learn a more general aggregation function g such that

$$\hat{y}_i = g(\beta_1(\mathbf{x}_i) h_1(\mathbf{x}_i), \dots, \beta_k(\mathbf{x}_i) h_k(\mathbf{x}_i)) \quad (1.10)$$

The authors used an overall architecture in which β and h are realized by deep neural networks because of their large modeling capacity. They call it a **Self-Explaining Neural Network** (SENN). If g is also differentiable, then the whole model can be trained using gradient descent and back-propagation. Figure 1.7 displays the whole architecture. For each prediction, an explanation is generated as a set of couples coefficient/concept (β_j/h_j) , as shown in Figure 1.8. While this approach enables to get coefficients that are tailored to each instance, its interpretability is not straightforward. In Figure 1.8, the coefficients are clearly presented but their corresponding concepts are ill-defined. Actually, concepts are only illustrated by their most representative instances. It makes them difficult to characterize and thus the interpretation may not be clear, especially for people who have no knowledge about the inner working of the model.

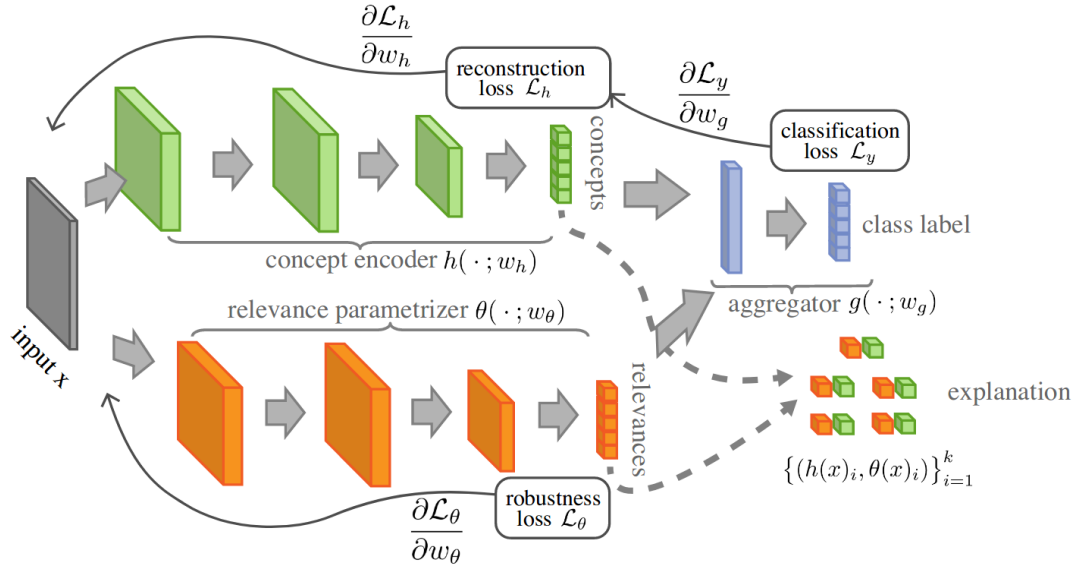


FIGURE 1.7: Overall architecture of the self-explaining neural network (Alvarez Melis and Jaakkola, 2018). There is a parametrizer (in orange) that generates relevance scores that correspond to β . Then, a encoder is used (in green) to learn higher level concepts from the raw features of the input space, which correspond to h . Finally, an aggregating layer enables to define g and an explanation is obtained as a set of couples coefficient/concept.

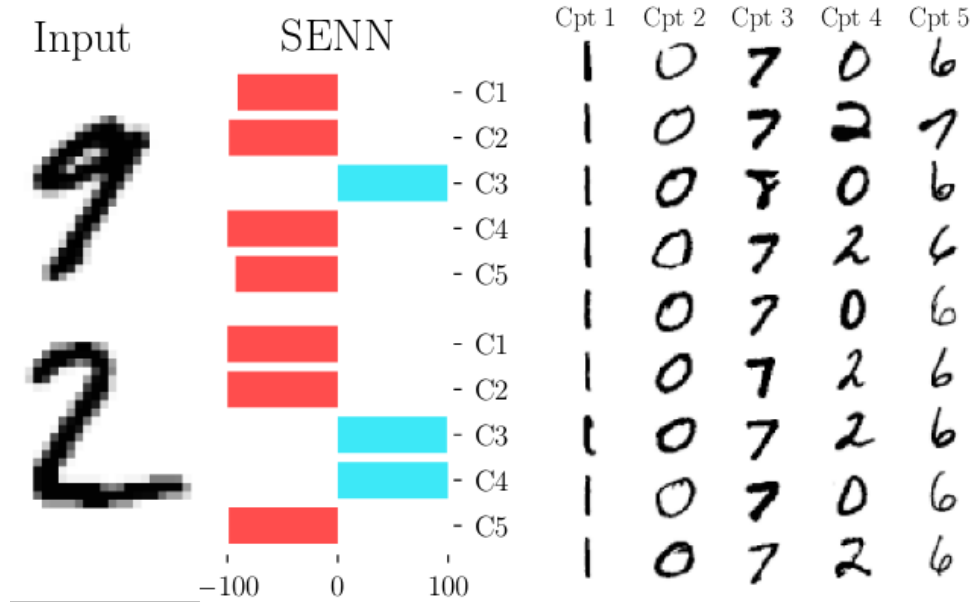


FIGURE 1.8: Example of explanation produced by a self-explaining network (Alvarez Melis and Jaakkola, 2018) on the MNIST dataset. Five concepts have been learnt and their corresponding coefficients are shown for two different inputs. For each concept, the most representative instances in the dataset are displayed.

(Li et al., 2018) proposed a deep-learning-based approach that consists in comparing in a latent space an instance with a set of prototypes, where each prototype is very close or identical to an instance from the training set (in the latent space). We call this approach **case-based reasoning through prototypes** (CBRP). The model is composed of two main parts:

- an autoencoder such that:
 - the encoder enables to reduce the dimensionality of the input and to learn more abstract features for prediction,
 - the decoder enables to visualize the prototypes that were learnt in the latent space.
- a prototype classification network: it learns several prototypes in the latent space and assesses the distances between the encoded input and each prototype to feed the fully-connected layers. The closest prototypes to the input can then be used as an interpretation.

The overall architecture of this network is shown in Figure 1.9. The advantages of this approach is that it is able to learn useful features unlike traditional case-based learning methods, prototypes can be visualized using the decoder network and the distances between an instance and all the prototypes are easily computable so that the classification can be interpreted. However, it is difficult to accurately characterize which concept a prototype represents. Also, the distance in the latent space may not be easily interpretable. As for SENN, this interpretation may not be useful for someone who does not have any knowledge about the model.

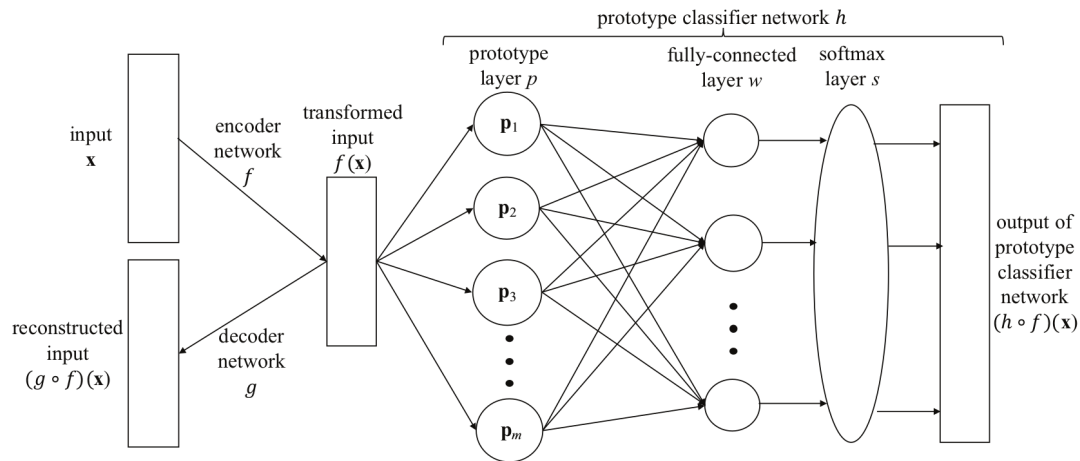


FIGURE 1.9: Architecture of the network proposed by (Li et al., 2018). Inputs are projected into a latent space using an encoder so that they can be compared to the prototypes that were learnt. The decoder enables to visualize these prototypes.

Another family of approaches relies on the properties of decision trees. As we saw in the previous section, they are well suited for explaining decisions.

(Alonso and Bugarín, 2019) proposed a web service aiming at providing users with local multimodal (textual + graphical) explanations. In particular, it uses fuzzy logic (Zadeh, 1965) to generate linguistic terms for each features, which will enable to generate a natural language explanation. Given a trained decision tree, at each

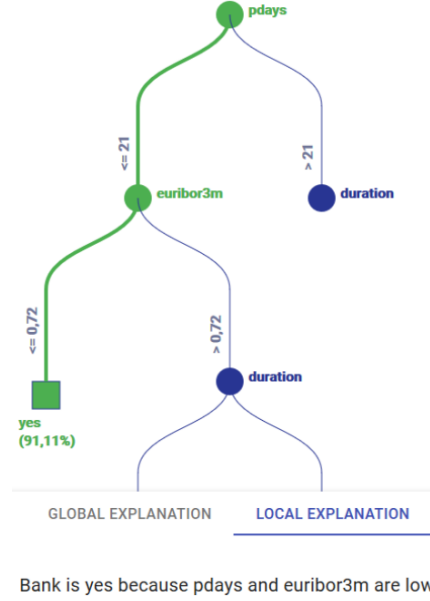


FIGURE 1.10: Example from (Alonso and Bugarín, 2019). The goal is to predict if, following telemarketing, a consumer subscribes a term deposit. This is represented by the variable *Bank* that can take two values: “yes” or “no”. Here, the branch leading to the prediction is highlighted (in green) and an explanation in natural language is generated. We can see that the splitting criteria on the features *pdays* and *euribor3m* are characterized by the linguistic term “low”.

of its node, the system computes the linguistic term that is the most similar to the splitting criterion. At the end, an explanation in natural language can be computed along the branch of the tree that led to the classification. An example is shown in Figure 1.10. Providing an explanation in natural language is particularly suited for people without any knowledge about the model being used. However, the name of the features is literally displayed in the explanation whereas their meaning is not clear. That shows that, without interpretable features, the explanation may not be understandable. A similar method is proposed for explaining the classification provided by a rule set.

(Lundberg et al., 2019) proposed a method called **TreeExplainer**, which computes the *exact Shapley values* (Shapley, 1953) in polynomial time for tree-based models. Let $\mathbf{x}_{i,|\mathcal{F}'}$ be an instance restricted to the features in $\mathcal{F}' \subseteq \mathcal{F}$ and $f_{|\mathcal{F}'}$ be the model’s output for a training set restricted to \mathcal{F}' . For a feature $F_j \in \mathcal{F}$ and the output of the model $f(\mathbf{x}_i)$, the Shapley value $\phi_j(f(\mathbf{x}_i))$ associated to F_j is:

$$\phi_j(f(\mathbf{x}_i)) = \sum_{\mathcal{F}' \in \mathcal{P}(\mathcal{F} \setminus \{F_j\})} \frac{|\mathcal{F}'|!(|\mathcal{F}| - |\mathcal{F}'| - 1)!}{|\mathcal{F}|!} (f_{|\mathcal{F}' \cup \{F_j\}}(\mathbf{x}_{i,|\mathcal{F}' \cup \{F_j\}}) - f_{|\mathcal{F}'|}(\mathbf{x}_{i,|\mathcal{F}'|})) \quad (1.11)$$

The first factor of the product is the proportion of possible combinations of F' in \mathcal{F} among all the possible combinations in \mathcal{F} . The second factor represents the contribution of the j -th feature to the output. Thus, $\phi_j(f(\mathbf{x}_i))$ can be interpreted as the average contribution of F_j to the prediction $f(\mathbf{x}_i)$. One big advantage of Shapley values is that only the output of the model is needed. However, the computational cost of evaluating all Shapley values is very high and only approximation methods are used in practice, except *TreeExplainer* that computes exact values. It works for

any tree-based machine learning model and relies on a local tracking of features in trees. The principle is to “*recursively keep track of what proportion of all possible subsets of features flow down into each leaves of the tree*”. We develop more on how Shapley values can be used to interpret models in Section 1.3.2.2, which is dedicated to local model-agnostic post-hoc interpretability approaches.

1.3.2 Post-hoc Interpretability

In this section, we focus on approaches that extract interpretations from an already trained model.

1.3.2.1 Global

This subsection is dedicated to global post-hoc interpretability methods.

Model-specific

Many post-hoc interpretability methods aim at evaluating how features contributed to the output. In particular, some of them are global model-specific approaches that enable to assess the role of features in the model (Breiman, 2001; Olah et al., 2017; Yosinski et al., 2015; Jakulin et al., 2005).

In the context of tree-based ensemble methods, (Breiman, 2001) proposed two different metrics to evaluate **feature importance**, which characterizes the contribution of a feature to the global behaviour of a model:

- The first one is usually called *permutation importance*. It is based on out-of-bag instances, which are instances that were not used for training one tree of the forest. For these out-of-bag instances, for each feature $F_i \in \mathcal{F}$, the values of F_i are randomly permuted. The importance is the difference between the accuracy obtained on the out-of-bag instances with and without permutation. The idea is that the performance should drop if the values of an important feature have been permuted among out-of-bag instances.
- The second metric is sometimes referred to as the *Gini importance*. For a given feature F_i , it stores the impurity decrease at each node where the splitting criterion was performed over F_i . It is weighted by the number of instances at that node. Then, it is averaged over all trees and normalized so that the sum of all importances is equal to 1.

These two approaches are performed after training. Permutation importance can be used for models relying on bagging while Gini importance can be measured for any tree-based ensemble model.

For deep neural networks, a first approach consists in performing **feature visualization** (Olah et al., 2017; Yosinski et al., 2015). It aims at visualizing the concepts that have been learnt by a unit of the neural network, such as a single neuron, a channel or a complete layer. The goal is to find the input that maximizes one such unit. A simple method would be to find in the dataset the instances that maximize the chosen unit, but what we may notice is just a correlation between those instances and the true behaviour of the model. A more interesting approach is to find the input that triggers the unit the most (Erhan et al., 2009). This optimization process is usually constrained by a regularization or diversity objective. An example of feature visualization is displayed in Figure 1.11. While it does not shed light on the global

reasoning of the model, it helps to understand what kind of patterns are detected by a specific unit of the network.

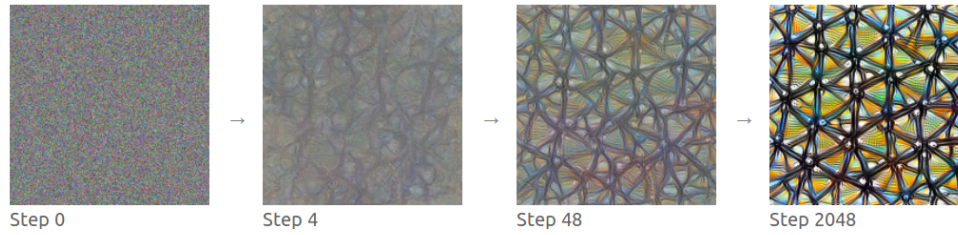


FIGURE 1.11: Optimization process that led to an image maximizing the activation of a single neuron (Olah et al., 2017). The initial image has been filled with random noise. At the end of the process, we can see the type of pattern that activates the neuron the most. In this particular case, the image maximizing the activation represents a regular pattern that seems to be a texture.

Model-agnostic

The most direct way to get a global insight into a black-box model is to train a *global surrogate* model in order to approximate its behaviour. The surrogate model must be a transparent model (cf. Section 1.3.1) so that it can provide explanations. More precisely, on the training set \mathcal{D} , the goal is to make the surrogate model replicate as well as possible the predictions of the black-box model. As a consequence, it is trained using the outputs of the black-box model and not the ground truth. In particular, several methods propose to approximate a model by a decision tree (Craven and Shavlik, 1996; Buciluă et al., 2006; Hinton et al., 2015; Frosst and Hinton, 2017; Wu et al., 2018; Zhang et al., 2019).

(Craven and Shavlik, 1996) proposed to train a decision tree that approximates the concepts represented in the black-box model. The training algorithm, called *TREPAN*, relies on querying the black-box model during the training process and is similar to C4.5 (Quinlan, 1993) (cf. Section 1.3.1.1). Moreover, a constraint on the number of internal nodes in the tree can be set to ensure the interpretability of the model. Later, inspired by *TREPAN*, **knowledge distillation** (Buciluă et al., 2006; Hinton et al., 2015) was proposed. It consists in transferring the knowledge of a teacher model to a more compact student model. Originally, the main motivation for performing model distillation was to get a smaller model that is much easier to deploy and less computationally expensive while approximating the performance of the bigger model well. Recently, and similarly to *TREPAN*, this has been used to get a more interpretable model that could explain the predictions of a black-box model (Frosst and Hinton, 2017).

Knowledge transfer from the teacher model to the student model is achieved by minimizing a loss function whose target is the distribution of probabilities provided by the teacher model. (Hinton et al., 2015) also showed that integrating ground truth information during the training of the student model leads to better performance. Thus, a term that represents the standard loss of the student model with respect to the ground truth is added to the previous loss function.

An example of distillation has been published in (Frosst and Hinton, 2017) where a neural network is distilled into a soft decision tree (Irsoy et al., 2012), which is a decision tree where, at each node, instances are redirected to children according to a probability distribution. The idea is to go from a neural network relying on hierarchical representations to a decision tree relying on hierarchical decisions. Figure 1.12

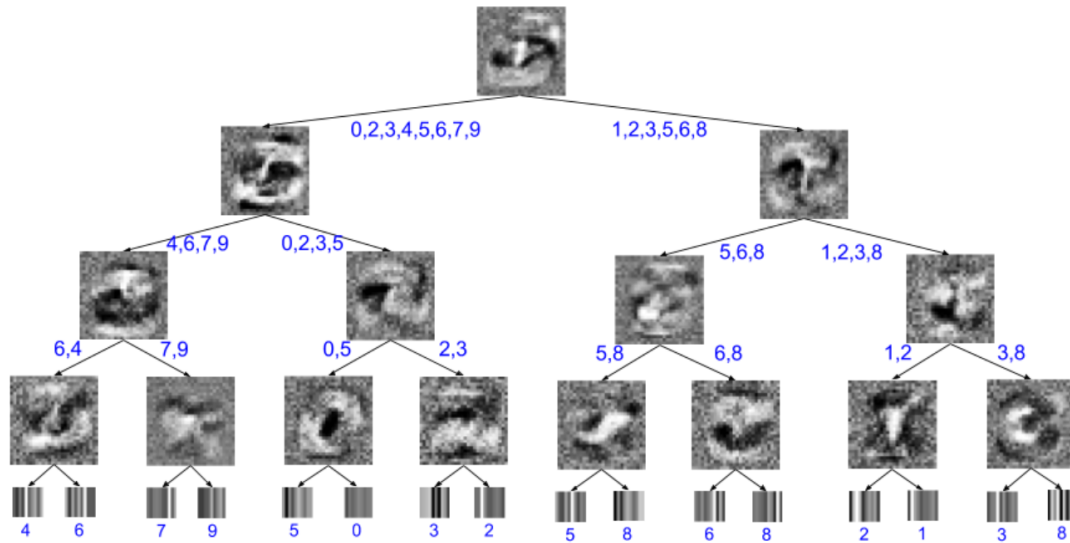


FIGURE 1.12: Example of a soft decision tree obtained by distillation of a teacher neural network on the MNIST dataset (Frosst and Hinton, 2017). The images at each node are the filters that were learnt to define the probability distributions used for splitting subsets of instances. The most likely classifications at each node and at each leaf are annotated in the tree.

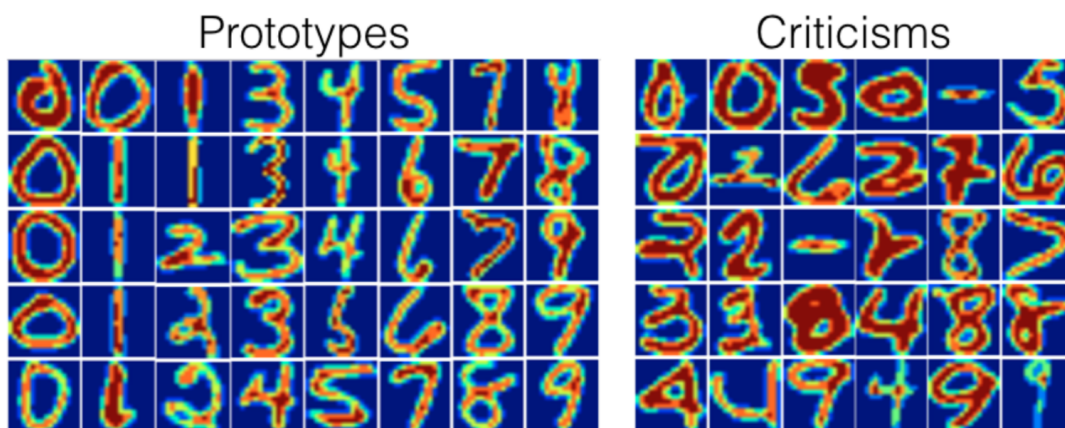


FIGURE 1.13: A set of prototypes and a set of criticisms that were learnt on the MNIST dataset (Kim et al., 2016). We can see that prototypes look more like handwritten digits that a human would expect than criticisms.

displays an example of a soft decision tree obtained by distillation of a teacher neural network on the MNIST dataset. Thus, the classification process is clearer since we can interpret each decision made by the tree. However, while the task is simple for a human (handwritten digit classification), an interpretation may be more complicated to understand since the concepts at stake in each node are only defined by their visual representation.

A type of approach that has been investigated recently aims at providing **example-based explanations**. A few specific instances, called *prototypes*, are used to explain the global behaviour of the model. *Fuzzy prototypes* have been proposed (Zadeh, 1982; Lesot et al., 2008) to perform classification and clustering. They are generated based on their similarity with instances from their class and on their dissimilarity with all other instances. *Case-based reasoning* (CBR) (Aamodt and Plaza, 1994), such as in (Kim et al., 2014; Kenny and Keane, 2019; Li et al., 2018), or *influence functions* (Koh and Liang, 2017) are also well suited for generating example-based explanations. However, prototypical instances are not sufficient when the dataset is not homogeneous enough (Kim et al., 2016). For this reason, (Kim et al., 2016) introduces the notion of *criticism* instances, which are data samples that do not fit the model well. Extracting prototypes and criticisms relies on the *maximum mean discrepancy* (MMD) (Lloyd and Ghahramani, 2015), which measures the difference between two distributions. Here, we measure the difference between the dataset \mathcal{D} and a subset $\mathcal{D}' \subseteq \mathcal{D}$. We look for the \mathcal{D}' that minimizes the MMD between \mathcal{D} and \mathcal{D}' to get a set of prototypes, and for the \mathcal{D}' that maximizes the MMD between \mathcal{D} and \mathcal{D}' to get a set of criticisms. In order to have relatively small subsets of prototypes and criticisms, the optimization process is performed under a size constraint on \mathcal{D}' . An example of a set of prototypes and a set of criticisms on the MNIST dataset is displayed in Figure 1.13. We can see that this type of approaches is well suited to images, but it does not explain the behaviour of a model. Experiments showed that the addition of criticisms to prototypes enable humans to classify the instances better. Thus, we can build with this approach a nearest neighbour (prototype) model for classifying instances. More interestingly, for a given trained model, we can specify its behaviour by analyzing its predictions on the prototypes and the criticisms. In particular, criticisms are instances for which the behaviour of the model may be unexpected since they represent data that is not well represented in the dataset. Also, if the dataset is big, it enables to quickly extract a few instances that should be harder to predict.

A more visual approach has been proposed by (Goldstein et al., 2015): **Individual Conditional Expectation** (ICE). The goal is to generate a plot that shows, for each instance, the evolution of the prediction with respect to one given feature (all other feature values being constant). Evaluations are made on a grid of values for the feature under study. This is actually an extension of the **Partial Dependence Plot** (PDP) (Friedman, 2001). Figure 1.14 shows an example of ICE and PDP.

1.3.2.2 Local

In this subsection, we focus on local post-hoc interpretability methods.

Model-specific

A few approaches proposed in the literature are based on the notion of **concept activation vector** (CAV) (Kim et al., 2018; Graziani et al., 2018; Ghorbani et al., 2019).

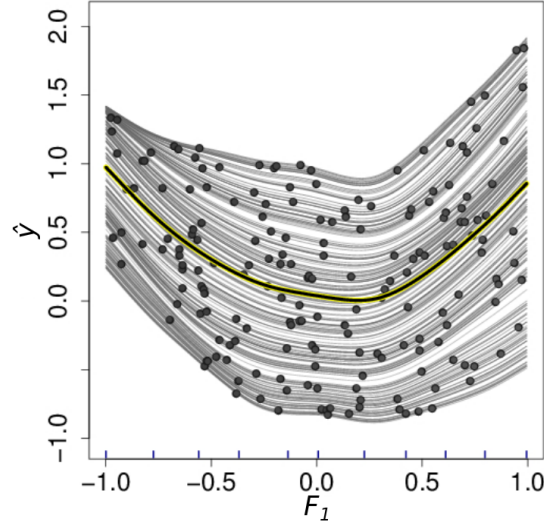


FIGURE 1.14: This plot represents the Individual Conditional Expectancy of a prediction with respect to feature F_1 (Goldstein et al., 2015). We can see that there is a parabolic relationship between the estimator \hat{y} and F_1 . The dots correspond to the actual value of F_1 for each instance. The yellow line is the Partial Dependence Plot, which is the average of the ICE over all instances.

This type of approach is tailored to deep neural networks and the goal is to compute the degree to which a user-defined concept contributes to a classification result. It relies on the four following steps (Kim et al., 2018), as illustrated in Figure 1.15:

1. The user needs to define a concept of interest (stripes in Figure 1.15) by feeding the system with a set of examples representing this concept. Those examples do not need to be part of the training set the model was trained on.
2. In a hidden layer l where higher level features have been extracted, a linear classifier is used in the space of activations of l to differentiate vectors representing the target concept from others. We can then define a concept activation vector \mathbf{v}_C^l as the normal to the hyperplane obtained with the linear classifier (oriented toward concept-based vectors).
3. For an instance \mathbf{x} , the conceptual sensitivity of class k to a concept C is computed as the following directional derivative:

$$\delta = \lim_{\epsilon \rightarrow 0} \frac{h_{l,k}(f_l(\mathbf{x}) + \epsilon \mathbf{v}_C^l) - h_{l,k}(f_l(\mathbf{x}))}{\epsilon} \quad (1.12)$$

with $f_l(\mathbf{x})$ the representation of \mathbf{x} in the space of activations of layer l and $h_{l,k}(f_l(\mathbf{x}))$ the probability that \mathbf{x} belongs to class k . It enables to assess the sensitivity of the predictions with respect to concepts at any layer of the network.

4. This is repeated for every instance of every class in the dataset to evaluate the impact of each concept on each class.

Thus, this approach is convenient for analyzing the sensitivity of a class to a specific concept. However, it requires expert knowledge to select examples of the concept under study. To solve this problem, (Ghorbani et al., 2019) proposed to perform a

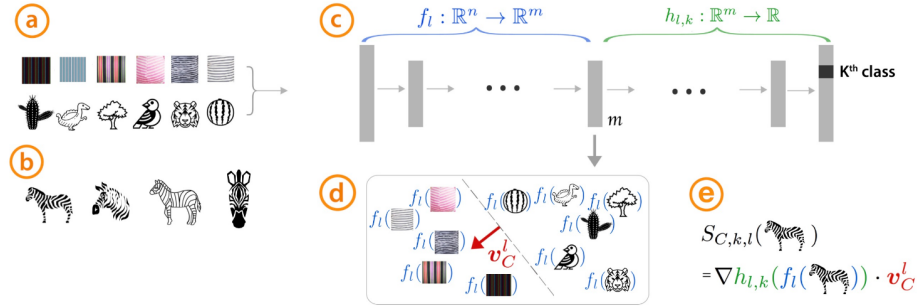


FIGURE 1.15: Architecture of the model presented in (Kim et al., 2018). In (a), the user provides a set of examples representing a concept (here “striped”) and a set of random examples. In (b), the training instances corresponding to the class under study (here zebras) are provided. In (c), f_l is the projection of an input instance into the latent space corresponding to layer l , and $h_{l,k}$ is the classifier. In (d), the concept activation vector \mathbf{v}_C^l is learnt by training a linear classifier to distinguish between the activations produced by the concept’s examples and the ones produced by other examples. Finally, in (e), the conceptual sensitivity is computed by deriving $h_{l,k}$ with respect to $f_l(\mathbf{x})$ in the direction of \mathbf{v}_C^l .

hierarchical segmentation of all the instances from a given class to use the segments as examples of concepts. These segments are then clustered in the activation space of layer l . Each cluster should represent a different concept and thus enables to define a concept activation vector. The conceptual sensitivity can then be computed as in the original approach. However, in this approach, automating the definition of concepts makes them lose any semantic interpretability.

An extension of the CAV for dealing with continuous concepts was presented in (Graziani et al., 2018). A linear regression is performed in the space of activations of layer l to seek the direction of greatest increase of the concept measures.

Inspired by image captioning methods, (Hendricks et al., 2016) proposed a neural-network-based approach for generating **visual explanations**. This work is based on the following hypothesis: a visual explanation should be both *discriminative*, which means that it should be class relevant, and *descriptive*, which means that it should be image relevant. The explanation produced by the model is a natural language sentence generated by a *long short-term memory* (LSTM) (Hochreiter and Schmidhuber, 1997). In order to be both class and image relevant, the LSTM is fed with compact features learnt with a convolutional neural network (CNN) and with the class label of the image that has been predicted by the CNN. The whole architecture is represented in Figure 1.16. The dataset used in this work was composed of images associated to five descriptive sentences (which are not explanations).

A similar multimodal explanation approach was proposed in (Huk Park et al., 2018). The model is trained on images that are all associated to natural language explanations (and not descriptions). It provides a textual explanation and highlights the areas in the image that point to the visual evidence for the classification. Since the model learns on images and their explanations, it does not need to explicitly distinguish between discriminative and descriptive information. However, it relies on a dataset where explanations were provided by humans. Such datasets are scarce and costly to obtain.

We saw in Section 1.3.2.1 that feature visualization is a global approach for getting insight into what a neural network does. At a local level, it is possible to go further by computing for each neuron visualization how much it has been activated

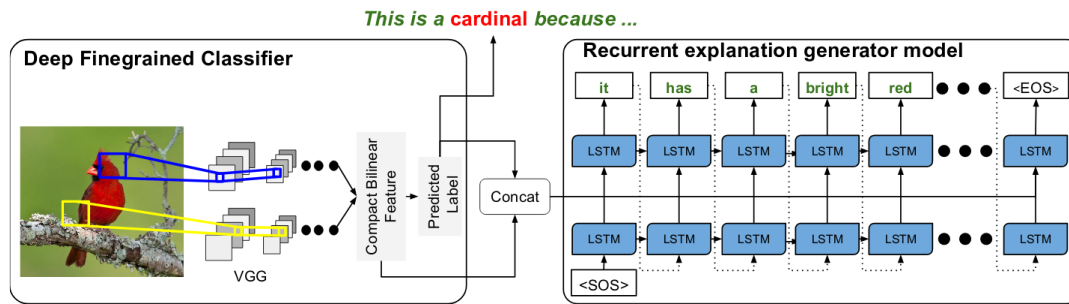


FIGURE 1.16: The model presented in (Hendricks et al., 2016) extracts visual features and predicts a class label. Then, sentence generation is conditioned by both the visual features and the class label to get a discriminative and descriptive explanation.

(Olah et al., 2018). That enables to detect to which extent a feature was detected at a particular position in the image.

However, although feature visualization allows to know what a network detects, it does not help to know how the detected features contributed to the final prediction. The latter can be obtained by performing **feature attribution**. The most common approach for getting this information with neural networks is to compute *saliency maps*. There are three types of methods for generating them (Kindermans et al., 2019):

- gradient-based methods (Simonyan et al., 2013), which analyze how a small change to the input affects the model's output,
- signal-based methods (Zeiler and Fergus, 2014; Springenberg et al., 2014; Kindermans et al., 2017), which isolate input patterns that fire neurons in higher layers,
- attribution-based methods (Montavon et al., 2017; Sundararajan et al., 2017), which compute the contribution of input features to the model's output. They differ from gradient-based methods because the sum of all contributions should be approximately equal to the output.

An example of saliency map from (Simonyan et al., 2013) is displayed in Figure 1.1 on page 11. While these methods are very useful for interpreting how a neural network produced a specific output, one should be very careful when using them:

- The saliency map itself is not interpretable without the input.
- (Kindermans et al., 2019) showed that some methods are completely fooled by adding a constant shift to the input data while this basic transformation does not have any impact on the output of the model.
- (Adebayo et al., 2018) demonstrated that some saliency methods are completely independent of both the model and the data generation process. Indeed, these methods produced the same saliency maps for trained and randomized models, and for correctly and randomly labelled data.
- Some feature attribution methods can be fooled by adversarial attacks (Zhang et al., 2018; Slack et al., 2019).

Feature attribution has also been extended by (Zhou et al., 2018) to deal with several concepts. Thus, one saliency map is computed for each concept, which enables to understand the contribution of each concept to the decision.

The last few years have also seen the emergence of *attention-based* models (Vaswani et al., 2017). These are neural networks that rely in particular on computing a mask that enables to characterize how important to the output the features generated by the model are. Thus, the values of the mask and their corresponding features may help to understand what the model focuses on, such as with feature attribution methods.

Model-agnostic

Local model-agnostic post-hoc interpretability methods enable to get an a posteriori interpretation that does not depend on the internal mechanism of the model. Since they focus on specific instances and not on the global behaviour of the model, they often look for an approximation of the model around the instance under study. This approximation is an explainable model, such as linear regression (Baehrens et al., 2010; Ribeiro et al., 2016), rules (Ribeiro et al., 2018; Guidotti et al., 2018; Pedreschi et al., 2019) or abductive reasoning (Ignatiev et al., 2019).

The most famous interpretability method is called **LIME**, which stands for Local Interpretable Model-agnostic Explanations (Ribeiro et al., 2016). LIME's principle is the following: given an instance x and its prediction by a black-box model, the local behaviour of the model is obtained by sampling instances around x , weighting them by their distance to x and training a Lasso on them (cf. Figure 1.17A). The loss function is the sum of a term representing how well the linear model locally approximates the black-box model and a term representing the complexity of the surrogate model (the number of coefficients in the case of Lasso). Then, the coefficients of the local linear model can be interpreted in the same way as presented in Section 1.3.1.1. Another interpretable model can be used, such as decision trees or rules.

The same author proposed an improvement of LIME actually relying on rules, which is called **anchors** (Ribeiro et al., 2018). An anchor is a *if-then* rule that covers an area of the decision space where the decision is (almost) always constant (precision criterion specified by the user). The type of area covered by an anchor is illustrated in Figure 1.17B. Knowing precisely where the anchor holds is an advantage on local linear surrogates (such as LIME) for which it is unclear whether they apply or not to an unseen instance. A similar approach was proposed in (Lakkaraju et al., 2019), where two-level decision sets (cf. Section 1.3.1.1) are used for characterizing a subspace of the feature space (first level) in which a set of rules represents the decision logic of the black-box model (second level). This approach also allows the user to specify a set of features on which the explanations, and so the rules, should be based.

Another popular post-hoc local model-agnostic approach is to perform feature attribution by computing **Shapley values** (Shapley, 1953; Štrumbelj and Kononenko, 2010; Lundberg and Lee, 2017). As we explained in Section 1.3.1.2, Shapley values represent the average contribution of a feature to the output of a model. However, computing the exact values is usually intractable (except for tree-based models with *TreeExplainer*, as shown in Section 1.3.1.2). This is why most algorithms aim at computing an approximation of these values. In particular, **SHAP** (SHapley Additive exPlanations) (Lundberg and Lee, 2017) is a popular approach that proposes efficient methods for computing the (approximate) Shapley values of any model on an instance. These methods are based on known methods such as LIME or DeepLift

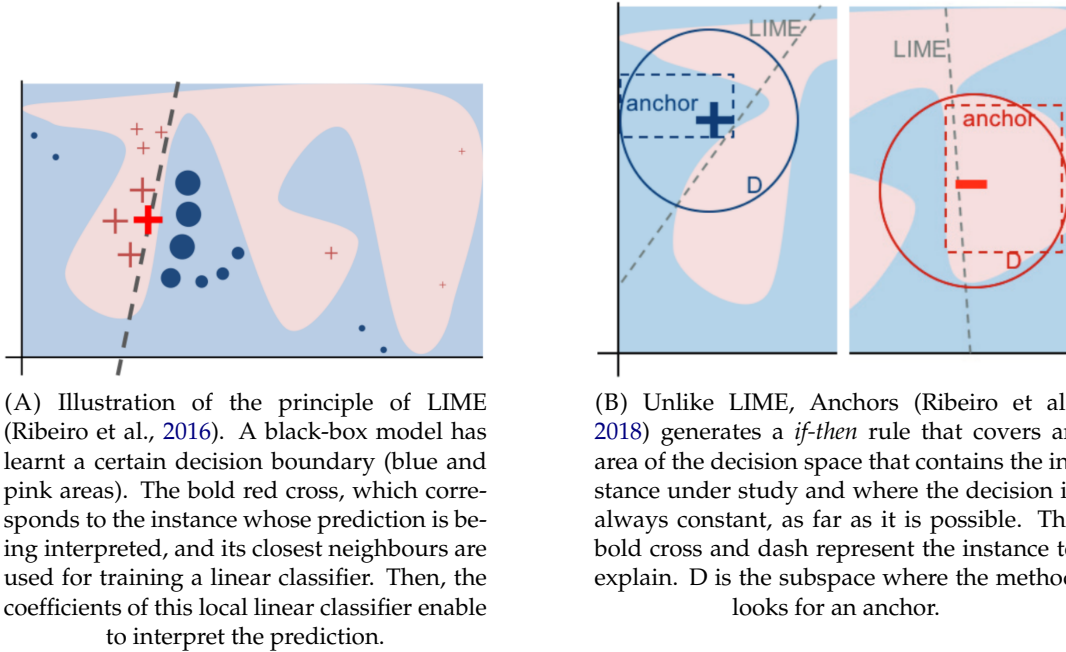


FIGURE 1.17: Illustration of the principles of LIME and Anchors.

(Shrikumar et al., 2017), which is why the authors claim to unify several XAI methods.

For an instance \mathbf{x} , the explanation model g to a model to explain f is proposed by SHAP as the following:

$$g(\mathbf{x}) = \sum_{j=0}^m \phi_j, \phi_i \in \mathbb{R} \quad (1.13)$$

where ϕ_j is the contribution of feature F_j to $g(\mathbf{x})$ for all $j \in \llbracket 1; m \rrbracket$ and ϕ_0 is the output of the model when all features have been removed. Attributions are computed using one of the methods proposed by SHAP to calculate Shapley values. Thus, SHAP can compute feature attributions for any type of models. However, as neural network feature attributions methods (cf. Section 1.3.2.2), SHAP (and thus LIME too) can be fooled by adversarial attacks (Slack et al., 2019).

(Chen et al., 2018b) proposed an approach called **L2X** that is, to the best of our knowledge, the only one *learning* (over the whole training set) how to compute feature attributions on a single instance. It is based on maximizing the *mutual information* between a subset of features and the output of the model. In particular, it is much faster than approaches like LIME and SHAP and it enables to perform *real-time* interpretation.

Another popular trend is to propose **counterfactual explanations**. A counterfactual explanation specifies what would have happened if an event had occurred differently. For example, “*If the car had turned on the left, it would have dodged the biker*” is a counterfactual explanation. From a psychological standpoint, such explanations have two main advantages (Byrne, 2019). First, counterfactuals favor causal judgments by amplifying the causal link between an action and its outcome. In our example, users’ judgments of a causal relation between the antecedent, turning on the left, and the outcome, the car dodging the biker, is amplified. Second, people make fast inferences from counterfactuals. In our example, people will understand

that the car did actually not dodge the biker as fast as with an affirmative sentence saying that the car did not dodge the biker.

For an instance (\mathbf{x}, y) , for a given $y' \in \mathcal{Y}$, for a decision function f and for a distance d defined on the input space, a counterfactual can be computed by finding (Wachter et al., 2017b)

$$\operatorname{argmin}_{\mathbf{x}'} \max_{\lambda} \lambda (f(\mathbf{x}') - y')^2 + d(\mathbf{x}, \mathbf{x}') \quad (1.14)$$

The counterfactual (\mathbf{x}', y') can be interpreted as the closest \mathbf{x}' to \mathbf{x} as possible such that $f(\mathbf{x}')$ is equal to y' . When λ is high, we lay more emphasis on getting an outcome close to y' , while a low λ favors close input features (\mathbf{x}' and \mathbf{x}). As we can notice, it is a local method that only needs the output of the model. Thus, this is an ideal candidate for interpreting black-box models and this is one of the reasons it has gained popularity. An application to images has also been proposed (Goyal et al., 2019).

However, a major drawback of this approach is that there may be many possible counterfactual explanations for a single instance prediction, depending on how different \mathbf{x}' is from \mathbf{x} . For instance, for a high-dimensional feature space, several feature changes could lead to a counterfactual explanation. So a strategy must be designed to select the best explanation.

A different area of research is the generation of explanations based on **knowledge graphs**, which are networks of real-world entities from one or several domains that define classes and relations between these entities (Paulheim, 2017). (Chen et al., 2018a) proposed such an approach for transfer learning explanations and (Lécué and Pommellet, 2019) focused on explaining object detection. The whole architecture of the model is displayed in Figure 1.18. Given a trained object detection model, contextual information is extracted from a set of publicly available knowledge graphs for each category of object the model can detect. This contextual information is a dictionary in which each key is an object label and its value is a subset of object labels that are linked to it. Then, for each detection, the confidence score of the model may be upgraded by looking at the context (other object detections in the image) and assessing how consistent it is with the contextual information previously extracted. The contextual predictions that contributed the most to the confidence increase are used for explaining the output of the model.

1.3.3 Recap

Table 1.1 contains the family of methods that are mentioned in bold font in this section. This enables to get an overview of how the different categories in our taxonomy are represented. We notice that there are few global post-hoc interpretability methods whereas local model-agnostic ones are much more common. Also, transparent models are well represented. Overall, there are several methods in every category, so one has to choose one of them by taking into account the type of model and the nature of the explanations.

1.4 Evaluating Explanations

When dealing with systems that put much emphasis on explainability, it is important to properly assess how pertinent explanations are in order to compare existing

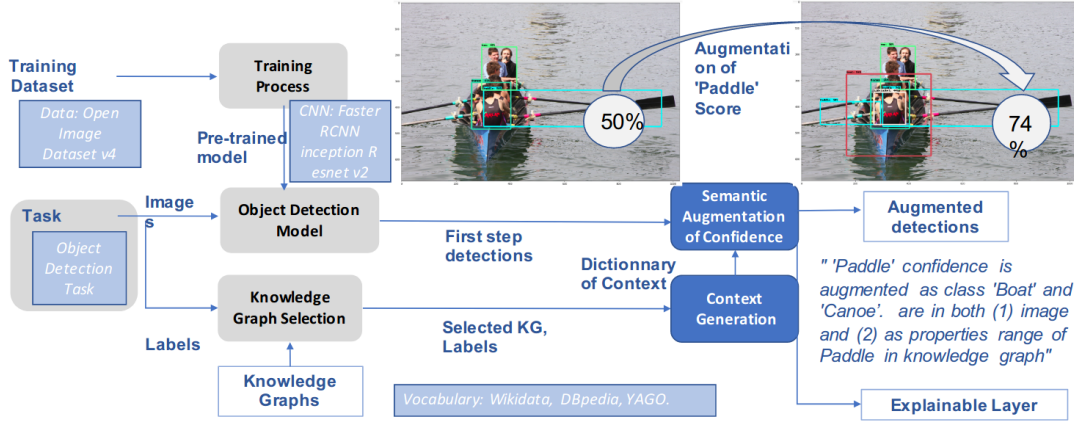


FIGURE 1.18: Architecture of the model presented in (Lécué and Pommellet, 2019). An object detection model is trained and applied on an image. In parallel, contextual information is extracted from knowledge graphs for each type of object in the dataset. Depending on the consistency between this contextual information and the predictions of the other objects in the image, the confidence of the output of the model increases or not.

	Transparency		Post-hoc interpretability	
	Global	Local	Global	Local
<i>Model-specific</i>	Decision rules Decision trees Linear regression Expert systems GLM/GAM	TreeExplainer SENN CBRP Naive Bayes kNN	Feature importance Feature visualization	CAV Feature attribution Visual explanations
<i>Model-agnostic</i>	X		Knowledge distillation Example-based explanations ICE/PDP	LIME Shapley values, SHAP Anchors Counterfactual explanations L2X Knowledge graphs

TABLE 1.1: Table representing the taxonomy we propose for XAI methods. The approaches classified in this table are described in Section 1.3. There is no example of method that relies on both transparency and model-agnosticism since, by definition, model-agnostic methods do not rely on any model.

explainable models and evaluate how efficient they are.

Let us make a few remarks about the example of explanation in Figure 1.1 on page 11. All the causes that are given in this explanation seem perfectly sound to us. But are they for all possible explainees? It actually depends on the explainee's knowledge. If we replace "It has a dog's nose" by "It has a nose", the explanation would still make sense. However, it would then also make sense for cat classification. This leads to the following question: is it acceptable to have the same explanation for two different results? Again, it depends on the end-user's knowledge and the goal of the system. Another question is: is the explanation short or long enough to be trusted? As we see, even a simple example can raise several questions. That is why assessing explanations is a complex task.

Miller and Hoffman *et al.* made reviews of the main factors that play a role in the

human assessment of a good explanation (Miller, 2017; Hoffman et al., 2018). The authors state that a good explanation needs to be *coherent*. That means that it must be consistent with the end-user's knowledge (Thagard, 1989). Humans prefer *simpler* explanations (those that cite fewer causes) and more *general* explanations (those that explain more events) (Read and Marcus-Newhall, 1993). Also, people do not usually judge an explanation on the basis of its *probability*, but rather on its *usefulness* and *relevance* (McClure, 2002). Vasilyeva et al. (Vasilyeva et al., 2015) show that the *goal* of the explainer and the *mode* of the explanation are both critical in its evaluation. In the end, there may be a trade-off between making an explanation more likely to be understandable, acceptable and trustable and making more likely explanations.

Several solutions have been proposed in the XAI literature (Doshi-Velez and Kim, 2017). The authors classify them into three families of methods:

- *Application-grounded* evaluation, where an expert directly evaluates how good an explanation is. This method is accurate but also time-consuming and an expert is required. It presents the same drawbacks as labeling instances in supervised learning.
- *Human-grounded* evaluation. Here, a human is asked to perform simple experiments that are still linked to the target. For example, one or several humans could be asked to select the best explanation among several of them. This is faster than application-grounded evaluation but also less accurate since the task to perform has been simplified. An example of human-grounded evaluation is proposed by (Baaj and Poli, 2019). They proposed a survey aiming at assessing 17 criteria, which are shown in Table 1.2. Each one of these criteria is evaluated using a Likert scale (Likert, 1932) like: strongly agree, agree, undecided, disagree, strongly disagree. In computer vision, another possibility is to compare human and model attention to assess how the model matches human behaviour (Das et al., 2017).
- *Functionally-grounded* evaluation. This consists in assessing the explanations of one model with another model that has been previously validated as an explainable model. This proxy model can be difficult to obtain since it must achieve completeness toward the original model (Gilpin et al., 2018). Therefore, an explainable local approximation of the model to explain is usually applied around the prediction (Ribeiro et al., 2016). Other approaches have been proposed to automatically assess visual explanations. In (Hendricks et al., 2018), similarly to (Hendricks et al., 2016), the strategy is based on both class and image relevance. Those two notions are not equivalent, as shown in Figure 1.19, and explanations may lead to a strong bias towards the class definition. The proposed model evaluates how the visual attributes mentioned in an explanation are grounded in the image, which enables to compute a score indicating how image relevant the explanation is. In (Zeiler and Fergus, 2014; Schlegel et al., 2019), a perturbation analysis is run on the input features to see if explanations remain consistent.

Choosing the right evaluation method then depends on what has been done before, how much time we have and how accurate the assessment should be. A thorough evaluation should be application-grounded, but it is not convenient for assessing different models since it takes a lot of time. That is a problem because comparing different models is going to be necessary in order to properly set a benchmark. For example, that is straightforward to compare the performance of different

Natural language	Human-computer interaction	Content and form
<ol style="list-style-type: none"> Overall, explanations are written in a correct English Conjugation choices are appropriate and adequate Grammatical form of sentences is satisfying 	<ol style="list-style-type: none"> Explanations are simple to use and easy to read Explanations help to make decisions faster than without Explanations let you change your opinion about your expectations Explanations help to take good decisions and are convincing Data and explanations are enough to trust the system Explanations express indirectly the way of the system is reasoning 	<ol style="list-style-type: none"> Length of explanations is adequate Explanations are repetitive It is difficult to read explanations until the end Content layout and order of elements in explanations are satisfying All causes are identified in explanations Explanations are sufficient in the sense that they do not contain superfluous information and do not miss one Overall, explanations seem consistent Explanations are true

TABLE 1.2: Criteria for evaluating explanations (Baaj and Poli, 2019). They are split into three categories: *natural language*, *human-computer interaction* and *content and form*. The first category aims at assessing the correctness of the language used in explanations. In the second category, criteria enable to evaluate what the explanation conveys when it is transmitted from the system to the user. The third category is dedicated to assessing the content and the form of the explanation.

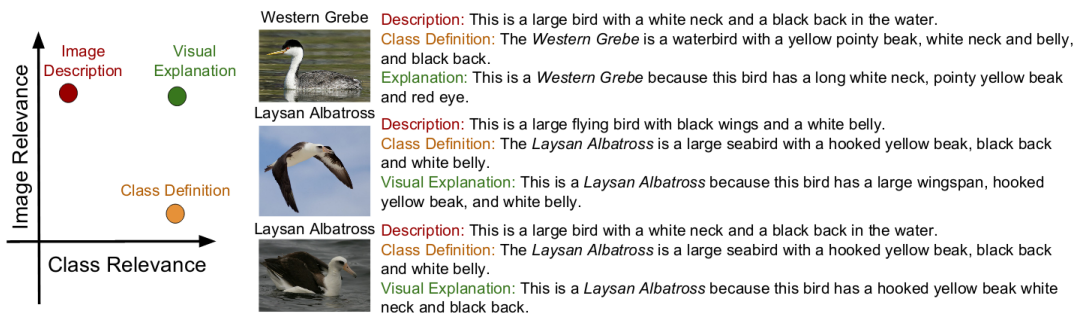


FIGURE 1.19: Example from (Hendricks et al., 2016) that presents the difference between image descriptions, which are not necessarily class relevant, and class definitions, which are not necessarily image relevant. A visual explanation should rely on class discriminative features that are also present in the image under study.

classifiers on a same test set using tools like confusion matrices. However, comparing explained classifications on this same test set without an expert would be much more complicated. Yet, it is necessary to compare models and methods, to set benchmarks or to certify a model. As there is no universal explanation evaluation metric, this task will be complicated to achieve. As of now, a convenient way to compare models would be to mix application-grounded, human-grounded and functionally-grounded evaluations. One model could be thoroughly assessed by an expert and then used as a landmark for evaluating other models using human-grounded or functionally-grounded evaluations.

1.5 Impact of Explanations on Users

Evaluating an explainable model or method is paramount, but it is also important to think about how an explanation affects the end-user. This is related to the notion of *mental model*, which we briefly present in this section. We also refer to a few works that experimentally studied how end-users are impacted by explanations.

A mental model is a mental representation of a specific environment based on knowledge and a description of this environment (Johnson-Laird, 1983). This is critical in XAI because, for a given explanation, a user will mentally represent the reasoning of the model to understand it. There can be several mental models associated to one explanation depending on its content. For instance, let us consider a problem where the goal is to learn the spatial relations between objects A, B, C, D and E (inspired from (Johnson-Laird, 2010)). We have learnt the following relations:

- B is to the left of A,
- C is to the right of B,
- D is below C,
- E is below B.

Then, two possible representations are:

B	A	C	and	B	C	A
E		D		E	D	

Those are two mental models of the spatial relations that were learnt. Thus, an explanation should make the end-user envision the mental models associated to the reasoning of the AI model under consideration.

Also, different explanations do not trigger the same reaction. (Lage et al., 2018) and (Booth et al., 2019) studied the various effects of explanations on the end-users. The authors carried out experiments where participants were asked to process different explanations given as logical statements. Participants had to either simulate or verify the prediction of the system given a set of inputs and an explanation, to say how confident they are in their decision or to determine if the decision changes when a perturbation slightly alters the input. (Booth et al., 2019) also proposed examples from three different domains (highway driving, emergency triage and chopsticks, which is a combinatorial hand game similar to tic-tac-toe). The authors mainly focused on participants' response times, which led to several observations. The more complex the explanation, the greatest the response time of the user. While that observation seems consistent with what most would expect, it is interesting to see the effects of different kinds of complexity:

- Increasing the number of rules and the number of terms in each rule makes the response time higher.
- Repeating variables has much less impact on response time than introducing new concepts.
- Participants took longer to process many simple rules than to process few long rules.
- Counterfactuals take longer to process, which may happen because participants try to imagine more mental models.
- The domain has a big impact on how well participants used the explanations to simulate the response of the model or how long they take to process explanations. One reason may be that different domains require domain-dependent explanations.

In these two experiments, we can notice that the three main sources of complexity in explanations are the introduction of new concepts or new domains and counterfactual reasoning.

(Poursabzi-Sangdeh et al., 2018) presented the results of an experiment where participants were shown functionally identical models (linear regression) with only two differentiating factors: the number of input features (2 or 8) and the model transparency (displaying the coefficients of the regression or not). Participants were asked to use these models to predict the prices of apartments in a single neighborhood of New York City. The first observation is that participants who were shown a transparent model with few input features simulated the model prediction better. However, increased transparency did not enable participants to detect the model mistakes as well as they did for a black-box model. The authors state that it is due to users being overwhelmed by the supplementary information provided by the transparent model. We argue that the form of the explanation also had an impact on these results since non-expert people (recruited on Amazon Mechanical Turk) may need a clearer explanation than what a linear regression model can provide.

(Papenmeier et al., 2019) investigated the impact of model accuracy and explanation fidelity on user trust. The task consisted in detecting offensive content in a dataset of tweets. Participants were presented three different models (from the most accurate to the least: a CNN, a logistic regression and a CNN trained on inverted labels) combined with three different types of interpretations (from the most reliable to the least: keyword highlighting using the L2X algorithm (Chen et al., 2018b), random keyword highlighting and no interpretation at all). The results show that model accuracy had the biggest impact on user trust. Then, while high-fidelity interpretations do not improve user trust, providing a low-fidelity interpretation decreases trust. The authors argue that further experiments with richer explanations are needed.

Overall, we come to the conclusion that the fidelity of an explanation and the way it is presented have a significant impact on the end-user. Thus, XAI applications should be user-dependent in order to help getting a better understanding of the model under study.

1.6 Discussion

In this chapter, we presented an extended view of the current field of XAI. We first started by proposing definitions of explainability and interpretability. This is very

important because there are at the moment no consensual definitions. Also, it is necessary for specifying our line of research. Then, we proposed a taxonomy that gives a clearer idea of the families of approaches that have been proposed. It is based on the three following criteria: transparent/post-hoc, local/global and model-agnostic/model-specific. This was the basis for our review of state-of-the-art methods. We also described how those approaches can be evaluated and how they impact users.

This work led us to observations that highlight the core steps of XAI design:

- Explainability is linked to rendering the reasoning of a model while interpretability consists in extracting elements that impacted a model prediction. This difference directly translates into two families of XAI approaches: transparent models, which offer a relatively clear reasoning, and post-hoc interpretability methods, which are post-training operations to gain insight into what a model does. Ultimately, choosing one or the other should depend on the application and on the user.
- Raw input features may not be the best elementary unit for building explanations (e.g. pixels in an image). Since all ML models ultimately rely on the features they are fed with, there must be an *anchoring* step that links original or constructed features to concepts that will be used in explanations.
- Explanation evaluation is one of the biggest hurdle in the field and few precise methods have been proposed. This is also user- and application-dependent, so it is tied to the XAI approach that is performed.
- The form of the explanation has a significant impact on how better users understand a model prediction. Thus, explanation properties such as its length or its clarity should all matter in the design of an XAI.

In the following chapter, relying in particular on the conclusions and observations of this chapter, we are going to present and specify the approach we propose.

Chapter 2

Proposed Approach

After having presented the current state of the field of XAI in the previous chapter, we introduce now our positioning. In particular, we focus on high-stake applications, which represent the cases where XAI is the most needed. In such applications, it is necessary to have a clear understanding of what a model does. Besides, it may rely on features that may not be easy to translate into understandable concepts, and vice versa. Those are two major requirements that the approach we propose must fulfill.

Based on our observations from Chapter 1, we are going to motivate the choices we made for designing an XAI able to perform explainable classification and annotation. In Section 2.1, we specify the type of explanations we would like our model to return. In Section 2.2, we precise the nature of the model we want to build. This is a critical point since the nature of the explanation and the transparency of the reasoning directly depend on it. Then, in Section 2.3, we detail our proposition for adding semantics in our model. Indeed, raw input features may not be understandable enough and, thus, should not contribute to make an explanation. Finally, we finish this chapter by giving an overview of the whole approach in Section 2.4.

2.1 Which Explanation?

One of our main goal is to build a model that can provide explanations for the decisions it returns. This leads to constraints that would not be considered if explainability were not a requirement.

The form of the explanations is very important because they are application- and user-dependent. Different end-users may not have the same requirements depending on their knowledge about the target task. Thus, the way the reasoning and the features are expressed to the user must be chosen carefully:

- It is not reasonable to expect that users without knowledge about ML will understand how the model works, no matter how transparent it is (Biran and McKeown, 2017). In that case, the end-user will probably be more comfortable with natural language explanations since they are expressed in a familiar formalism.
- The end-user may not have the knowledge to perform the target task. Thus, providing interpretations such as saliency maps or concept activation vectors may not make the model more trustworthy.
- For features, qualifying linguistic expressions may be more satisfactory than numeric values in an explanation (Biran and McKeown, 2017; Michalski, 1983).

To fulfill these three points, we would like to have a model that produces explanations as sentences in natural language. This ensures that an explanation will be understandable regardless of the knowledge of the actual user.

2.2 Which Model?

As we saw in the previous chapter, there is a wide range of XAI models with distinct characteristics. There are two main approaches for explaining the behaviour of a model: either building a model that is inherently transparent, or extracting an interpretation from a model after training. While a transparent model enables to track its own reasoning, any post-hoc interpretability method cannot be completely faithful to the model under study (otherwise the original model would not even be needed) (Rudin, 2019).

Transparent models may require more time to build, especially for adding domain expertise. Indeed, feature engineering may be required for the model to handle features that are understandable to humans and specific to the target application. For high-stake applications, this extra effort outweighs the advantages brought by black-box models, which may be flawed or too complex (Rudin, 2019).

As a consequence, we decided to rely on transparency to ensure that the generated explanations are truly representative of the reasoning performed by the model. In Section 1.3.1.1 on page 14, we mentioned three families of globally transparent models: rule-based models, decision trees and linear models. Rules and decision trees are closer to human reasoning (Byrne and Johnson-Laird, 2009) and to the language of reasoning (Pedreschi et al., 2019), which is logic. However, decision trees are known to be unstable (Turney, 1995; Dwyer and Holte, 2007) because they can produce very different models for small changes in a dataset. So our strategy will rely on learning classification rules, which present the advantage of being transparent and explainable. For annotation, we propose to learn in a similar way constraints that will be then used to solve a *constraint satisfaction problem* (CSP) (Vanegas et al., 2016). This constraint-based approach for annotation also satisfies the transparency requirement.

2.3 Which Features?

Besides transparent reasoning, explainable models also rely on features that can be translated into understandable concepts. This particular characteristic is very important because if the features handled by a transparent model are not understandable to humans, explanations will not be either.

In this thesis, we focus on applications in which the input is a signal, such as an image or a time series. For other types of input, features are usually associated to a meaningful variable. In the case of signals, raw input features are not the best fit for producing human-understandable explanations since they do not correspond to what human understanding is based on (Ribeiro et al., 2016; Alvarez Melis and Jaakkola, 2018). For instance, a pixel is the elementary unit to digitally represent a 2D image, but it is not the elementary unit in human image understanding. More generally, the signals that are provided to ML models are digital and are thus different from the signals a human being processes. Therefore, it is important to extract more abstract representations from these raw input features.

Higher-level features represent properties or, more globally, relations that convey contextual information. For example, in images, scene description and understanding relies on the analysis of spatial relations between entities in the input image (Freeman, 1975; Biederman, 1981). The classification of (multivariate) time series is also based on patterns that are extracted from the signal and that represent relationships among input features (Geurts, 2001). The same observation can be made for text understanding where a word may have a different meaning depending on how it is linked to other words in a sentence. Thus, extracting semantic relations is paramount for understanding and interpreting a signal.

While deep neural networks are able to learn complex relationships in the feature space, those are not represented in a form adapted to the generation of explanations. Approaches have been proposed for detecting visual relationships in images (Dai et al., 2017; Lu et al., 2016; Donadello et al., 2017) or for performing statistical relational learning (Dumancic et al., 2019a) which aims at predicting relations between instances. However, those are not necessarily transparent and they rely on big datasets, which is not always compatible with critical applications. (Clément et al., 2018) proposed an approach for learning spatial relations in images without requiring any expert supervision. It is based on force histograms (Matsakis, 2002) and generates a hierarchical spatial descriptor. However, while relations can be interpreted by looking at the histograms, there is no straightforward way to qualify them in terms that are understandable to humans.

Instead of providing knowledge about relations by labeling instances, which is very time-consuming, we propose to define once, before training, a catalogue of potentially relevant relations that we call *vocabulary*. This vocabulary should be set beforehand by an expert to ensure that it is suited to the target task. The idea is to learn on training instances relations from the vocabulary rather than learning from already annotated relations in instances. This enables to deal with a much wider range of datasets and inputs. It also implies a different learning strategy. Relations from the vocabulary have to be assessed on the training set and then only the most relevant of them are extracted for building the model. This type of strategy has already been performed in evolutionary fuzzy systems (Fernandez et al., 2019; González et al., 2012). However, it can only deal with input features that are already anchored in human-understandable concepts. The approach we propose goes further and is able to cope with more complex inputs.

2.4 The Overall Approach

Based on the choices that we made in previous sections, we detail here the approach we propose. The goal is to build a classification or annotation model by learning relations of interest from a given vocabulary on a training set. The explanations will be based on these learnt relations.

In Section 2.2, we chose to build rules for classification and constraints (to define and solve a CSP) for annotation. Rules are usually inferred on the feature space but, as we wrote earlier, input features may not be suited to build explanations on. This is why we propose to extract relations that are associated to linguistic descriptions. A convenient framework for representing such relations is *fuzzy logic*, which enables to take into account both qualitative and quantitative information (Zadeh, 1965; Zadeh, 1999). Also, there exists an extensive literature about spatial and temporal fuzzy relations, as shown in Chapter 7 on page 105. This makes the whole framework well-suited for managing images and time series.

Relations from the vocabulary are assessed on entities that are part of the instances in the training set. Those entities are either directly provided in the dataset we use or we have to extract them using a segmentation algorithm.

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

1. The relations from the vocabulary \mathcal{V} are assessed on the training set \mathcal{D} . In Chapter 5 on page 71, we present two strategies to prevent unnecessary computations to make the evaluation process faster.
2. The most relevant relations are extracted according to the learning algorithm presented in Chapter 4 on page 55.
3. Rules or constraints are generated from the relevant relations so that classification or annotation can be performed. They can then be translated into a natural language explanation using the linguistic description associated to each relation in the rules/constraints, as shown in Chapter 6 on page 87.

The whole approach is illustrated in Figure 2.1 in the case of image annotation.

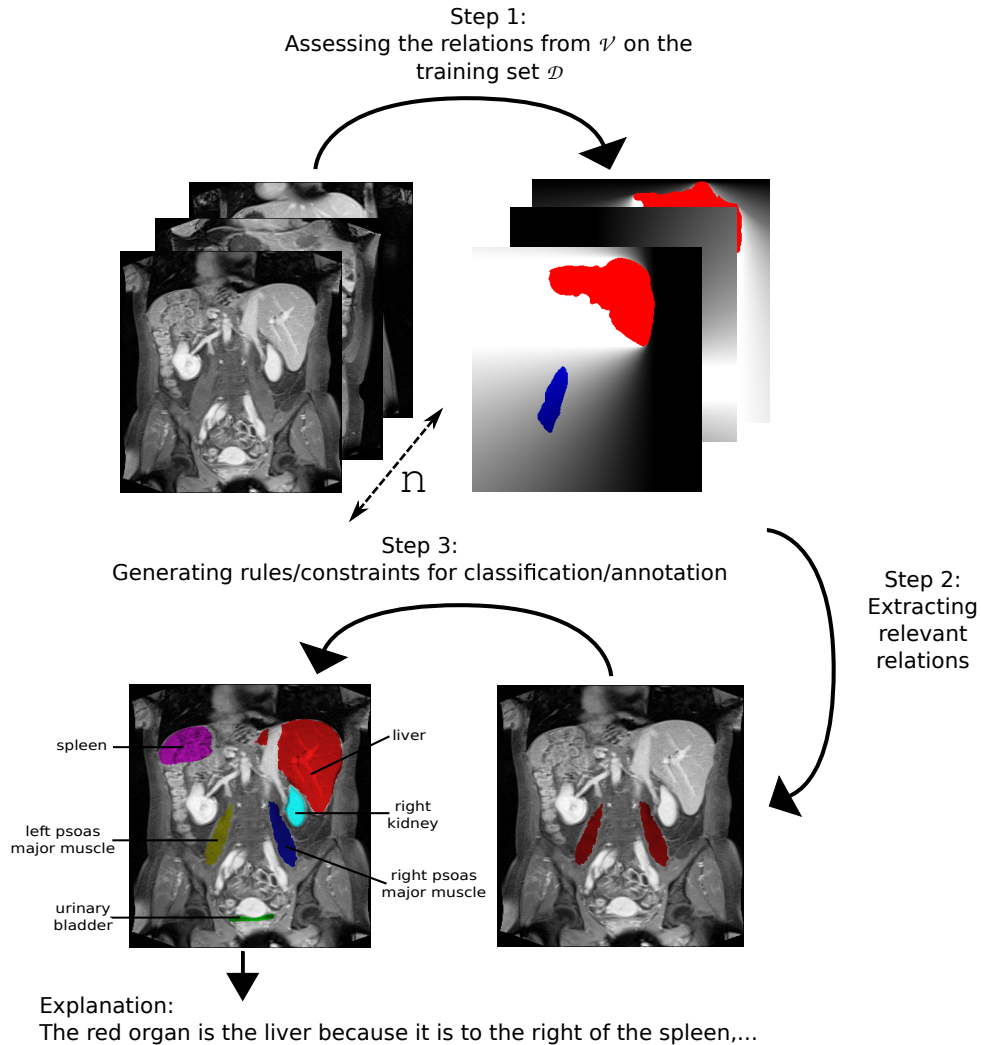


FIGURE 2.1: Illustration of the approach proposed in this thesis in the particular case of image annotation. \mathcal{V} is a vocabulary of relations and \mathcal{D} is the training set. For each annotation, an explanation is provided based on the constraints that directly led to this result.

The next part of this thesis consists in four chapters that are dedicated to presenting more precisely this approach and our contributions.

Part II

Building an Explainable Model

Introduction

We present in this part the approach we introduced in Chapter 2. It aims at building a transparent model able to both tackle classification or annotation problems and provide explanations in natural language to its results. As we saw in Chapter 1, few approaches return such explanations. The visual explanations proposed by (Hendricks et al., 2016; Huk Park et al., 2018) fit this requirement but their model is not transparent, which is not suited for critical applications. (Alonso and Bugarín, 2019) proposed to generate explanations in natural language from a decision tree. Although it relies on fuzzy logic, it handles features that are not necessarily understandable.

The principle of our approach is based on the fact that a human quickly learns how to describe and recognize a concept on just few instances using his/her knowledge. Indeed, this is possible because core features of this concept occur consistently among instances while irrelevant features do not (Kellogg, 1980). Thus, we propose to learn relevant features from few data based on their frequency in the training set. We assume here that the features we look for are present in most instances.

The features our approach relies on are relations between entities in the instances of the training set. These entities can be, for example, an object or a part of it in an image, or one signal in a multivariate time series. Since we need understandable features, we decided to handle *fuzzy* relations. Those are expressed within the fuzzy logic framework, which enables to express both quantitative and qualitative information and to manage the imprecision of the language and of the data. Thus, many fuzzy relations from the literature are associated to a linguistic variable (Zadeh, 1975), which makes them interpretable.

In this thesis, we deal with classification and annotation tasks. We consider that classification consists in assigning a label to an instance. As we wrote in Chapter 2, we would like to build rules to perform this task. As for annotation, in the scope of this work, we consider it as the problem of assigning a label to different entities in an instance. We would like to achieve that by solving a fuzzy constraint satisfaction problem (FCSP).

Learning class by class the most frequent subsets of fuzzy relations enable to get class-specific descriptors. This can be achieved once the relations from a given vocabulary have been evaluated on the training set. Then, a frequent subset of relations can be turned into a rule (classification) or into a set of constraints (annotation). This modeling is transparent, which enables to understand how relations are used for making a decision. Since the fuzzy relations we use are associated to a linguistic variable, rules or constraints can be translated into natural language to generate an explanation.

In this part, we first present in Chapter 3 the characteristics that make our approach expressive. It relies on a vocabulary of relations and the use of the fuzzy logic framework. Then, we focus in Chapter 4 on describing how class-specific relation-based descriptors are extracted using a fuzzy frequent itemset mining method. As the approach requires to evaluate many different relations, including compute-intensive

ones, Chapter 5 deals with the strategies that have been developed to prevent unnecessary computations during the evaluation of relations. This chapter aims at making the evaluation, and thus the training, faster. Finally, Chapter 6 tackles the solutions we propose for solving a classification or an annotation problem and explaining it. They are based on building rules and generating constraints so that the model is transparent and an explanation can be expressed as a natural language sentence.

Chapter 3

Model Expressivity and Fuzzy Relations

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

The expressivity of a model is linked to the diversity of situations it can express. This depends on:

- The features it relies on. The more features, the more expressive the model could be. However, the expressivity of the explanations depends on the number of understandable features.
- The type of reasoning the model can perform. In the context of explainability, the reasoning of the model should be quickly understandable by a human, which is linked to the notions of transparency and simulatability presented in Section 1.2 on page 11.

While the second point has been previously tackled, we will specify here the features we use.

In the present chapter, we first focus on defining the notion of expressivity that was introduced in the previous paragraphs. We review how it has been defined in other fields and then we specify it in the context of our approach. Then, in the second section, we present how fuzzy relations, which are the features we use here, contribute to the expressivity of the approach. They rely on *fuzzy logic*, which is a form of many-valued logic that is well suited to the problem we tackle. We introduce this framework and specify the formalism of fuzzy relations.

3.1 Expressivity of a Model

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

works refer to it as *expressiveness* or *expressive power*), which has already been defined in several other fields.

There exist several works dealing with the *expressivity* of a language. In knowledge representation, the notion was introduced in the 1980's in an informal way (Levesque and Brachman, 1987). Later, Baader (Baader, 1996) gave a formal definition of the expressive power of knowledge representation languages. This definition states that two knowledge representation languages have the same expressive power if and only if one language can be expressed by the other and vice versa. While this enables to compare the expressive power of two different knowledge representation languages, it does not define formally the expressive power of one knowledge representation language. Borgida (Borgida, 1996) built on Baader's work to compare the expressive powers of description logics and predicate calculus. The comparison is based on the *meaning*, which is defined for both languages and represents all the possible interpretations of a given description or set of predicates. We emphasize that, in this context, an interpretation is a mapping from a description or set of predicates to a subset of the domain of values it is working on.

From a machine learning point of view, expressiveness is often mentioned in the deep learning community but it is rarely defined. Cohen et al. (Cohen et al., 2016) define it as the space of all possible configurations of parameters of the network. Raghu et al. (Raghu et al., 2017) proposed an approach to measure the expressive power of neural networks. They define the expressivity as the influence of the architecture of a neural network over the resulting functions it computes.

In this work, we define expressivity similarly to the definitions from description logics we presented above. It should reflect the diversity of situations the model can generate. As we presented in the previous chapter, in the approach we propose, the model is given a catalog of relations that we call *vocabulary*. A richer vocabulary should lead to a more expressive system, which should help to produce better decisions and explanations.

For example, if we classify objects based on their shape and size, a vocabulary containing only shape properties will entail a model that cannot correctly deal with the size. In this case, the vocabulary restrains the expressive power of the model too much. On the other hand, adding size properties to the vocabulary would enable the model to be more expressive. If the added properties are relevant to the problem under consideration, then the model should achieve better performance. However, enriching the vocabulary also leads to a less tractable model. Overall, an expressive model should rely on the same vocabulary a human would use for performing and describing the task.

Let us introduce the following notations:

- Let $\mathcal{V} = \{\mathcal{R}_1, \dots, \mathcal{R}_{n_V}\}$ be the vocabulary given for building a model. It is a set of n_V relations.
- Let $\alpha : \mathcal{V} \rightarrow \mathbb{N}$ be a function such as $\alpha(\mathcal{R})$ denotes the arity of the relation \mathcal{R} for each \mathcal{R} in \mathcal{V} .
- Let \mathcal{X} be the space where instances are defined.
- Let \mathbf{x} be an instance defined on the space \mathcal{X} .
- Let $\mathcal{O}_{\mathbf{x}} = \{o_{\mathbf{x},1}, \dots, o_{\mathbf{x},K} \mid o_{\mathbf{x},i} \in \mathcal{X}, \forall i \in \llbracket 1; K \rrbracket\}$ be a set of K entities in \mathbf{x} that are defined on \mathcal{X} . We assume here that the number K of entities is fixed.

- Let $E_x(\mathcal{V}) = \{\mathcal{R}(o_{x,1}, \dots, o_{x,\alpha(\mathcal{R})}) \mid \mathcal{R} \in \mathcal{V}, (o_{x,1}, \dots, o_{x,\alpha(\mathcal{R})}) \in \mathcal{P}(\mathcal{O}_x)\}$ be the set of all the relations in \mathcal{V} evaluated on the entities in \mathcal{O}_x . $\mathcal{P}(\mathcal{O}_x)$ is the power set of \mathcal{O}_x .

$\mathcal{P}(E_x(\mathcal{V}))$ is the set of all the possible explanations. It characterizes the expressivity of the model.

Property 1

The number of relations to evaluate for instance \mathbf{x} is:

$$|E_x(\mathcal{V})| = \sum_{j=1}^{n_{\mathcal{V}}} \frac{|\mathcal{O}_x|!}{(|\mathcal{O}_x| - \alpha(\mathcal{R}_j))!} . \quad (3.1)$$

Proof.

For each relation $\mathcal{R} \in \mathcal{V}$, we would like to compute \mathcal{R} for all the $\alpha(\mathcal{R})$ -ary subsets of entities included in \mathcal{O}_x . Since we do not make any assumption on the symmetry of \mathcal{R} , this is equivalent to obtaining the ordered subsets of k elements from a set of n elements. The number of such subsets is given by (Uspensky, 1937)

$$\frac{n!}{(n-k)!} .$$

Thus, the number of times a relation \mathcal{R} has to be evaluated is

$$\frac{|\mathcal{O}_x|!}{(|\mathcal{O}_x| - \alpha(\mathcal{R}))!} .$$

We get $|E_x(\mathcal{V})|$ by summing the contributions of each relation. □

For example, for an instance \mathbf{x} such as $\mathcal{O}_x = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ and a vocabulary $\mathcal{V} = \{\mathcal{R}_{\text{greater}}, \mathcal{R}_{\text{prime}}, \mathcal{R}_{\text{even}}\}$ such as $\alpha(\mathcal{R}_{\text{greater}}) = 2$, $\alpha(\mathcal{R}_{\text{prime}}) = 1$ and $\alpha(\mathcal{R}_{\text{even}}) = 1$, we get

$$|E_x(\mathcal{V})| = \frac{9!}{(9-2)!} + \frac{9!}{(9-1)!} + \frac{9!}{(9-1)!} = 90 .$$

The time complexity for computing all the evaluations thus depends on the factorial of the number of entities that is generated by \mathcal{O}_x . (Levesque and Brachman, 1987) stated first that there is a dependency between the expressive power of a knowledge representation language and its computational tractability. We encounter here a similar issue and we will propose two methods for reducing the evaluation space in Chapter 5.

3.2 Fuzzy Relations

We focus now on the type of relations the model handles: *fuzzy relations*. They are based on Fuzzy Logic, which presents several advantages:

- it enables to take into account both qualitative and quantitative information,
- it can manage the imprecision of the data and the vagueness of the language,

- the notion of linguistic variable has been defined to characterize a variable with natural language expressions.

These assets make fuzzy relations well-suited for computing with words (Zadeh, 1999) and thus for generating explanations in natural language.

In this section, we first remind briefly the core definitions of the fuzzy logic framework before dealing with fuzzy relations. A more extensive recap of the main notions in fuzzy logic is given in Appendix B on page 171.

3.2.1 Fuzzy Logic

Fuzzy Logic and the fuzzy set theory have been introduced by (Zadeh, 1965). It can be seen as an extension of Boolean logic that enables to manage imprecision. While a value is either true or false in Boolean logic, it can range from 0 (false) to 1 (true) in Fuzzy Logic.

Definition 3: Fuzzy Set

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

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

We will also rely on the notion of *core* of a fuzzy set.

Definition 4: Core of a Fuzzy Set

The *core* of a fuzzy set F defined on a universe \mathcal{U} is a non-fuzzy set defined as

$$\text{core}(F) = \{u \in \mathcal{U} | \mu_F(u) = 1\} . \quad (3.2)$$

We then define what a *linguistic variable* is (Zadeh, 1975). It is an important concept since it enables to give interpretability to fuzzy sets.

Definition 5: Linguistic Variable

A *linguistic variable* is defined as a triplet (V, Δ_V, F_V) such as:

- V is the name of the variable,
- Δ_V is the domain on which V is defined,
- $F_V = \{F_1, F_2, \dots\}$ is a finite collection of fuzzy sets. Each of these fuzzy sets is associated to a linguistic term which qualifies V .

For example, let us consider a linguistic variable (“duration”, $[0; 100]$, $\{F_{\text{short}}, F_{\text{long}}\}$) represented in Figure 3.1. Δ_V is here the domain of time in seconds. The fuzzy set F_{short} enables to characterize how short a duration is while F_{long} evaluates how long it is. For $\delta \in [0; 25]$, $\mu_{\text{short}}(\delta) = 1$ so a duration taking δ seconds is short. For $\delta \in [25; 75]$, the shortness is imprecise. The values of μ_{short} enable to quantify the vagueness of the definitions. For $\delta \in [75; 100]$, $\mu_{\text{short}}(\delta) = 0$ so the duration is not short. However, that does not necessarily mean that it is long. Indeed, there can be several intermediary linguistic terms in F_V between “short” and “long”, which would contribute to make the approach more expressive (at the cost of tractability).

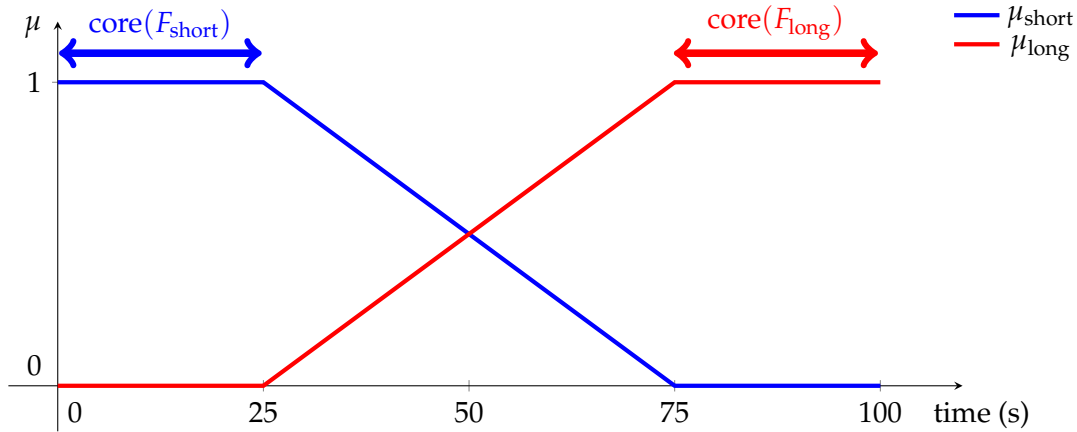


FIGURE 3.1: Figure representing the linguistic variable ("duration", $[0; 100]$, $\{F_{\text{short}}, F_{\text{long}}\}$). The membership functions of the two fuzzy sets F_{short} and F_{long} are represented respectively in blue and red. The core of F_{short} is $[0; 25]$ and the core of F_{long} is $[75; 100]$.

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

3.2.2 Fuzzy Relations

The fuzzy logic framework is also more suited to express relations between two sets than Boolean logic because it can express a wider variety of situations (cf. Figure 3.2).

Definition 6: Fuzzy Relation

Given two universes \mathcal{U} and \mathcal{W} , a binary fuzzy relation \mathcal{R} is characterized by a mapping defined as

$$\mu_{\mathcal{R}} : \mathcal{U} \times \mathcal{W} \rightarrow [0, 1] \quad . \quad (3.3)$$

It assigns a degree of relationship to any $(u, w) \in \mathcal{U} \times \mathcal{W}$. p -ary fuzzy relations are defined identically. A property can be seen as a unary relation, so we will only talk about relations in the remaining of this thesis.

In Figure 3.2A, the disk is perfectly to the left of the square. In such a situation, a membership function $\mu_{\text{to the left of}}$ would return 1 and a Boolean relation would return *true*. However, in Figure 3.2B, the situation is more difficult to evaluate. In the fuzzy case, we get a membership degree between 0 and 1, which is similar to the way a human would describe it. The Boolean logic is not adapted to handle cases like this one.

In the following, for the sake of comprehension, the word *relation* refers either to a fuzzy relation \mathcal{R} or to the degree of relationship $\mu_{\mathcal{R}}(u, w)$ between u and w . For instance, for a dyadic fuzzy relation $\mathcal{R}_{\text{to the left of}}$ and two objects u and w , we call relation the result $\mu_{\mathcal{R}_{\text{to the left of}}}(u, w)$, which represents the spatial relation " u to the left of w ".

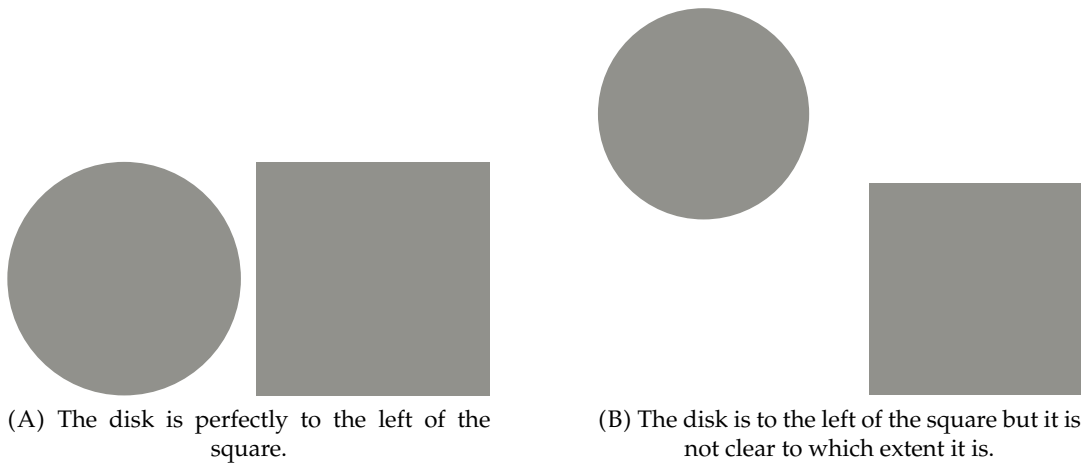


FIGURE 3.2: Two figures where the relation *disk to the left of square* cannot be expressed the same way. Fuzzy logic provides tools to characterize differently these two situations while relying on the same fuzzy relation *to the left of*.

3.3 Discussion

In this chapter, we proposed a definition of expressivity in the context of our approach. This was motivated by the fact that an expressive enough model is necessary to perform well and produce convincing explanations. We then reminded the main definitions from the fuzzy logic framework, which is convenient for generating explanations in natural language. We also illustrated the notions of linguistic variable and of fuzzy relation.

We saw that the number of relations to evaluate increases quickly with the number of entities in an instance. Therefore, there may be many relations to assess over a training set of several instances. In order to keep our approach tractable, we investigate two different ways:

- We propose in Chapter 5 two heuristics that aim at preventing unnecessary evaluations. The first one is based on a learning algorithm that we present in the next chapter. The second heuristic takes advantage of a few properties of relations, such as their symmetry or the logical implications between them.
- The second solution is relation-specific. We worked on the optimization of a computationally expensive type of relations so that they can be evaluated faster. This contribution is presented in Chapter 8.

In our approach, relations are evaluated on the entities in an instance. We assumed that the number of entities is fixed. This assumption can hold true if we are given the entities beforehand or if we know a priori the number of entities we are looking for. Clustering or segmentation methods could be applied too but are out of the scope of this thesis.

In the next chapter, we will see how to extract the most relevant relations among those that were evaluated.

Chapter 4

Learning Relevant Relations and Descriptors

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

The objective of this chapter is to propose a generic method for extracting relevant descriptors of the different classes that will be involved in a task of classification or annotation. We introduce an algorithm based on fuzzy frequent itemset mining that selects the most relevant relations in a vocabulary for representing a class. This algorithm takes as input a set of evaluated fuzzy relations between entities for each instance in the training set.

In Section 4.1, we introduce our hypothesis about describing classes by their most relevant relations. Section 4.2 presents the field of frequent itemset mining while Section 4.3 describes its fuzzy counterpart. Finally, Section 4.4 tackles the solution we propose for extracting the most relevant subsets of relations.

4.1 Relation-based Descriptors

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

In this thesis, we would like to determine the relations that are the most relevant to classes of entities. Thus, they should be representative of a class. However, that does not mean that they necessarily help to perform classification or annotation. For instance, the property that a tangerine is orange is relevant for classifying tangerines from bananas whereas it is not for classifying tangerines from oranges. While this property is important in describing the class *tangerine*, it is not sufficient for classification. As such, it should be part of a higher-level structure that represents a class and that is unambiguous. In other words, this structure should be both *descriptive* and *discriminative* (Hendricks et al., 2016; Lesot et al., 2008).

In this chapter, we focus on the extraction of *descriptors* for each class. We will describe how to turn them into discriminative structures in Chapter 6. A descriptor is a set of relations that describes a class. So, given the vocabulary and the expressivity of the approach, we would like to extract the most relevant sets of relations for describing the classes we deal with.

As we stated in Chapter 2, we do not rely on a statistical relational learning approach to extract these sets of relations. Such approaches usually work on relational data (Dumancic et al., 2019b), which is expensive to obtain since relations between entities need to be labeled. Another type of approaches relies on *propositionalisation* (Kramer et al., 1998) to turn a relational representation of a problem into a propositional one where each attribute is mapped to a value. This is what we are going to do here.

While regular datasets (where entities are labeled but not the relations between them) are also costly to label, they are much more common. In order to take advantage of them, similarly to (Zucker and Ganascia, 1996), we propose to bring the knowledge about relations beforehand by providing a vocabulary \mathcal{V} of potentially relevant relations. Thus, we can evaluate relations from \mathcal{V} on the training set and then extract the most relevant of them without relying on any labeling of the relations in the training instances.

We propose to rely on *frequent itemset mining* to extract the most frequent descriptors in the training set. **We assume here that relevance and frequency are equivalent.** This is a strong assumption that is justified by the fact that relevant features should occur consistently whereas irrelevant features should occur inconsistently (Kellogg, 1980). In other words, entities from one given class should share the same relevant relations. As a consequence, these relations should be frequent among the observations of entities from this class. The limits of this assumption are that learning on few instances could be highly impacted by the presence of one or several outliers. So we will have to be careful about the way the training set is built. Also, we extract sets of relations that are frequent together, which means that one set represents conjunctions of relations. Expressing only conjunctions may limit the performance of the model we build.

Since we are looking for the frequent subsets of relations of each class, we decide to carry out the learning phase using a *one vs all* approach. The learning is thus performed class by class.

4.2 Frequent Itemset Mining

As we saw in the previous section, and since we handle fuzzy relations, relevant descriptors are extracted using a *fuzzy frequent itemset mining* approach. So, in our case, it consists in performing the mining on the set of the fuzzy relations that were evaluated on the training set.

In this section, we present the field of frequent itemset mining to introduce the main concepts that will be used in *fuzzy frequent itemset mining*.

4.2.1 Background

Frequent itemset mining aims at extracting frequent patterns in a database. It has originally been introduced for performing *association rule learning* (Agrawal et al., 1993). In such problems, the goal is to build rules that catch the frequent patterns in the database. The most common example of association rule learning is the *market basket problem*. In this problem, we have a dataset of transactions made by customers. Each transaction contains items that one customer purchased, as represented in Table 4.1. Based on this dataset, the objective is to extract rules that describe well the behaviour of consumers, such as “🍷 \Rightarrow 🍷”. It relies on assessing which items are frequently bought together.

The association rule learning process can be divided into two steps:

1. finding all the frequent subsets of items in the database,
2. building rules based on the frequent subsets of relations.

The first step actually consists in frequent itemset mining, on which we focus here. It is the most computationally expensive step. It is applied on a database that can be represented as a *formal context*.

Definition 7: Formal Context (Ganter and Wille, 1996)

A *formal context* is a tuple $(\mathcal{T}, \mathcal{I}, \mathcal{R})$ such as:

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

For example, in the database $\mathcal{D}_{\text{market}}$ (cf. Table 4.1), we have $\mathcal{T} = \{t_1, t_2, t_3, t_4, t_5, t_6\}$ and $\mathcal{I} = \{\text{🧀}, \text{🍌}, \text{🍉}, \text{🍇}, \text{🍓}\}$. The corresponding formal context is represented in Table 4.2.

Transactions	Items
t_1	🧀 🍌 🍓
t_2	🧀 🍌 🍉 🍇
t_3	🍉 🍇 🍓
t_4	🧀 🍌 🍉 🍇 🍓
t_5	🍌 🍉 🍓
t_6	🧀 🍌

TABLE 4.1: Example of database for the *basket market problem*. We call this database $\mathcal{D}_{\text{market}}$.

Transactions	Items				
	🧀	🍌	🍉	🍇	🍓
t_1	1	1			1
t_2	1	1	1	1	
t_3			1	1	1
t_4	1	1	1	1	1
t_5		1	1		1
t_6	1	1			

TABLE 4.2: Representation of the formal context associated to $\mathcal{D}_{\text{market}}$ (cf. Table 4.1). Zeros are not displayed for a better visualization of the formal context.

In the remainder of this thesis, a subset of items will be called an *itemset*. We also introduce the notion of *k*-itemset.

Definition 8: k -itemset

For $k \in \mathbb{N}$, a k -itemset is a set of k items.

In order to assess whether an itemset is frequent or not, one first has to compute the frequency of this itemset in the database, which is called the *support*.

Definition 9: Support

The *support* of an itemset is defined as:

$$\forall I \subseteq \mathcal{I}, \text{support}(I) = \frac{|\{t \in \mathcal{T} \mid \forall i \in I, \mathcal{R}(t, i) = 1\}|}{|\mathcal{T}|} \quad (4.1)$$

The support of an itemset $I \subseteq \mathcal{I}$ ranges from 0 (if I is not a subset of any transaction) to 1 (if I is a subset of every transaction). For example, in $\mathcal{D}_{\text{market}}$ (cf. Table 4.1), we have $\text{support}(\{\text{🍇}\}) = \frac{3}{6} = 0.5$ and $\text{support}(\{\text{🍌}, \text{🍌}\}) = \frac{4}{6} \approx 0.667$.

Once the support of an itemset is known, it is compared to a threshold to determine whether it is frequent or not.

Definition 10: Frequent Itemset and Minimum Support

Let S be a real number in $[0; 1]$. An itemset $I \subseteq \mathcal{I}$ is said to be *frequent* if, and only if, $\text{support}(I) \geq S$. S is a threshold that is called the *minimum support*.

For instance, in $\mathcal{D}_{\text{market}}$, for $S = 0.6$, $\{\text{🍌}, \text{🍌}\}$ is frequent whereas $\{\text{🍇}\}$ is not.

Some of the frequent itemsets have interesting properties that make them ideal candidates for generating other frequent itemsets.

Definition 11: Maximal Itemset (Uno et al., 2004)

A frequent itemset $I \subseteq \mathcal{I}$ is said to be *maximal* if, and only if, it has no frequent superset.

For example, for $S = 0.6$, $\{\text{🍌}\}$ is frequent because $\text{support}(\{\text{🍌}\}) \geq S$ but it is not maximal because it is a subset of $(\{\text{🍌}, \text{🍌}\})$, which is also frequent. However, $\{\text{🍌}, \text{🍌}\}$ is maximal since it has no frequent superset. Intuitively, a maximal itemset could be used to generate smaller frequent itemsets that are actually its subsets.

Some other itemsets with interesting properties are the *closed* itemsets, which rely on a *closure operator*.

Definition 12: Closure Operator

A *closure operator* on a poset (cf. Appendix C on page 175) (A, \leq_A) is a function $h : A \rightarrow A$ such as

- $\forall a \in A, a \leq_A h(a)$,
- $\forall a_1, a_2 \in A, a_1 \leq_A a_2 \Rightarrow h(a_1) \leq_A h(a_2)$,
- $\forall a \in A, h(a) = h \circ h(a)$.

Definition 13: Closed Itemset (Pasquier et al., 1999)

Let h be a closure operator over $(\mathcal{I}, \leq_{\mathcal{I}})$. An itemset $I \in \mathcal{P}(\mathcal{I})$ is said to be *closed* if, and only if, $h(I) = I$.

In order to use those closed itemsets, we must define a closure operator. Such operators are usually built using a *Galois connection*.

Definition 14: Galois connection

Let (A, \leq_A) and (B, \leq_B) be two posets (cf. Appendix C on page 175). Let $f : A \rightarrow B$ and $g : B \rightarrow A$ be two functions defined over these two posets. The couple (f, g) forms a *Galois connection* between (A, \leq_A) and (B, \leq_B) if f and g are order-reversing and verify

$$\forall a \in A, \forall b \in B, a \leq_A g \circ f(a) \text{ and } b \leq_B f \circ g(b) \quad (4.2)$$

For example, the functions

$$\begin{aligned} f : \mathcal{P}(\mathcal{T}) &\rightarrow \mathcal{P}(\mathcal{I}) \\ T &\mapsto \{i \in \mathcal{I} \mid \forall t \in T, \mathcal{R}(t, i) = 1\} \end{aligned} \quad (4.3)$$

and

$$\begin{aligned} g : \mathcal{P}(\mathcal{I}) &\rightarrow \mathcal{P}(\mathcal{T}) \\ I &\mapsto \{t \in \mathcal{T} \mid \forall i \in I, \mathcal{R}(t, i) = 1\} \end{aligned} \quad (4.4)$$

form a Galois connection between $(\mathcal{P}(\mathcal{T}), \subseteq)$ and $(\mathcal{P}(\mathcal{I}), \subseteq)$. One advantage of these functions is that their composition forms a closure operator.

Lemma 1

Let (A, \leq_A) and (B, \leq_B) be two posets. Let $f : A \rightarrow B$ and $g : B \rightarrow A$ be two functions defined over these two posets. If (f, g) forms a Galois connection, then $f \circ g$ and $g \circ f$ are closure operators over (B, \leq_B) and (A, \leq_A) respectively.

Proof.

According to Definition 12, $g \circ f$ has to satisfy three conditions:

- $\forall a \in A, a \leq_A g \circ f(a)$ by definition of the Galois connection,
- $\forall a_1, a_2 \in A$ such as $a_1 \leq_A a_2$, we have $g \circ f(a_1) \leq_A g \circ f(a_2)$ since f and g are both order-reversing,
- $\forall a \in A, a \leq_A g \circ f(a)$, so $f(a) \geq_B f \circ g \circ f(a)$ since f is order-reversing. By definition of the Galois connection, $f(a) \leq_A f \circ g \circ f(a)$ so $f(a) =_A f \circ g \circ f(a)$. Thus, by composition, $g \circ f(a) =_A (g \circ f)^2(a)$.

Thus, $g \circ f$ is a closure operator over (A, \leq_A) . The reasoning is the same for proving that $f \circ g$ is a closure operator over (B, \leq_B) . \square

Using the definitions of f and g in Equation (4.3) and Equation (4.4) respectively, $h = f \circ g$ is thus a closure operator over $(\mathcal{I}, \leq_{\mathcal{I}})$. In the following, we will use this operator as our closure operator on \mathcal{I} .

For example, in $\mathcal{D}_{\text{market}}$, $h(\{\text{🍪}, \text{🍌}\}) = \{\text{🍪}, \text{🍌}\}$ because:

- $g(\{\text{🍪}, \text{🍌}\}) = \{t_1, t_2, t_4, t_6\}$,
- $f(\{t_1, t_2, t_4, t_6\}) = \{\text{🍪}, \text{🍌}\}$,
- $h(\{\text{🍪}, \text{🍌}\}) = f \circ g(\{\text{🍪}, \text{🍌}\}) = \{\text{🍪}, \text{🍌}\}$

so $\{\text{🍪}, \text{🍌}\}$ is a closed itemset.

4.2.2 Frequent Itemset Mining Algorithms

In this subsection, we provide a taxonomy of frequent itemset mining algorithms that is based on the work of (Fournier-Viger et al., 2017). There are four categories of frequent itemset mining algorithms:

- *breadth-first search* algorithms such as Apriori (Agrawal et al., 1993). They first look for the frequent 1-itemsets. Then, they generate the candidate 2-itemsets. Frequent 2-itemsets are extracted and we then focus on 3-itemsets and so on until no more itemset can be generated,
- *depth-first search* algorithms such as Eclat (Zaki, 2000). They first associate every 1-itemset I to its *tidset*, which is the set of transactions in which the corresponding itemset is included. Thus, the support of I is easy to compute since it is the cardinality of its tidset. Then, all possible 2-itemsets are generated and their tidsets can be obtained by intersection. The process continues until no more itemsets can be generated. It is usually faster than breadth-first search algorithms,
- *pattern growth* algorithms such as FP-Growth (Han et al., 2000). They first create a compact representation of the database as a tree. Then, the tree is used to generate all the frequent itemsets. It has the advantage of not generating candidate frequent itemsets,
- *reduction* algorithms that work on a compact representation of the frequent itemsets such as *closed* itemsets (Pasquier et al., 1999) or *maximal* itemsets (Uno et al., 2004). After the compact representation is generated, this family of methods relies on one of the previous categories of algorithms to generate the frequent itemsets.

In our case, we would like to mine frequent subsets of relations class by class. Thus, there is one different sub-database for each class. As instances from the same class should share some frequent relations, they should be highly correlated to each other. Since a closure operator working on an itemset I returns the set of items that are common to the transactions in which I belongs to, algorithms relying on closed itemsets should take advantage of this high correlation (Pasquier et al., 1999). However, since we deal with fuzzy relations, we need an algorithm that is able to manage such data.

4.3 Fuzzy Frequent Itemset Mining

Fuzzy frequent itemset mining relies on the same principles as frequent itemset mining. In this section, we update the previous definitions to deal with fuzzy data. Then, we will give a brief review of fuzzy frequent itemset mining algorithms.

It should be noted that **these algorithms still return crisp itemsets**. However, the relation between transactions and items is now fuzzy.

4.3.1 Fuzzy Formal Context and Support

The definition of a formal context needs to be updated since we work with fuzzy data. In that case, it is called a *fuzzy formal context*.

Definition 15: Fuzzy Formal Context (Belohlávek, 2012)

A fuzzy formal context is a tuple $(\mathcal{T}, \mathcal{I}, \mathcal{R})$ such as:

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

An example of fuzzy formal context is represented in Table 4.3.


Transactions	Items				
					
t_1	0.8	0.1	0.9	0.8	0
t_2	0	0.3	0.2	0	0.9
t_3	1	0.7	0.7	1	0.6
t_4	0	0.2	0	0.2	1
t_5	0.9	0.6	0.8	1	0.9

TABLE 4.3: A fuzzy database $\mathcal{D}_{\text{fuzzy}}$ represented as a fuzzy formal context.

Since the relation between transactions and items is now fuzzy, we also need to update the definition of the support of an itemset.

Definition 16: Support

The *support* of an itemset is defined as

$$\forall I \subseteq \mathcal{I}, \text{support}(I) = \frac{\sum_{t \in \mathcal{T}} \min_{i \in I} \mathcal{R}(t, i)}{|\mathcal{T}|} \quad (4.5)$$

For example, in $\mathcal{D}_{\text{fuzzy}}$ (cf. Table 4.3), $\text{support}(\{\text{cheese}, \text{carrot}\}) = \frac{1.4}{5} = 0.28$.

4.3.2 Fuzzy Frequent Itemset Mining Algorithms

Many fuzzy frequent itemset mining algorithms rely on the Apriori algorithm (Agrawal et al., 1993). Thus they are breadth-first search algorithms. The F-APACS algorithm (Au and Chan, 1998) first converts data into linguistic terms using the fuzzy set theory. A statistical analysis is performed to automatically set the minimum support threshold. (Kuok et al., 1998) proposed a different approach to handle quantitative databases for generating fuzzy association rules. They introduced a significance factor that reflects not only the number of transactions supporting an itemset, but also its degree of support. The FDTA algorithm (Hong et al., 1999) proposes another way of converting quantitative data into linguistic terms. In other algorithms, each feature is actually converted into several linguistic terms whereas the FDTA algorithm keeps for each feature only the linguistic term of maximal cardinality. That means that there are as many linguistic terms as features, which enables to reduce computational costs.

A completely different way of mining fuzzy frequent itemsets relies on pattern growth algorithms. The first step consists in fuzzifying data, if necessary. Then, the

tree is constructed and the final step is the mining of fuzzy frequent itemsets based on the previously constructed tree. (Papadimitriou and Mavroudi, 2005) proposed an algorithm called fuzzy frequent pattern tree (FFPT). Non frequent 1-itemsets are removed from the database and each transaction is sorted according to the membership value of its frequent 1-itemsets. Then, the tree is constructed by handling each transaction one by one. Since transactions are sorted by membership values, several different paths may represent the same itemset. As a consequence, a few useless tree nodes are generated. The compressed fuzzy frequent pattern tree (CFFPT) algorithm solves this problem by using a global sorting strategy (Lin et al., 2010b). However, this solution leads to attaching an array to each node, which requires more memory. (Lin et al., 2010a) proposed the upper bound fuzzy frequent pattern tree algorithm (UBFFPT). It estimates the upper bound membership values of frequent itemsets to avoid attaching an array to each node. This algorithm requires four database passes to build the tree. Then, the tree is parsed several times to generate all candidate frequent itemsets. Depending on the database, the tree can be deep and have a large amount of nodes. An ultimate database pass is performed to compute the support of each candidate frequent itemset. Unlike crisp pattern growth algorithms, these need to generate candidate frequent itemsets.

To the best of our knowledge, there is no fuzzy frequent itemset mining algorithm that relies on the generation of closed itemsets. As previously mentioned, this kind of algorithm is well suited to the data we deal with since it takes advantage of the high correlations between instances of the same class. This is why we worked on the fuzzification of such an algorithm.

4.4 Fuzzy Close Algorithm

In this section, we are going to present the algorithm we use for mining frequent subsets of fuzzy relations. It is based on an algorithm called *Close* (Pasquier et al., 1999), which we present in the first section. Then, we describe the fuzzy transposition of this algorithm and how we deal with fuzzy data.

4.4.1 The Close algorithm

The Close algorithm has been proposed by (Pasquier et al., 1999). It can only handle crisp formal contexts. It finds all the frequent closed itemsets so that it can work on a more compact representation of the frequent itemsets. Since there are often less frequent closed itemsets than frequent itemsets, the search space is smaller, the computation is less costly and the number of database passes is reduced. The algorithm relies on the following properties (Pasquier et al., 1999):

1. all subsets of a frequent itemset are frequent;
2. all supersets of an infrequent itemset are infrequent;
3. all closed subsets of a frequent closed itemset are frequent;
4. all closed supersets of an infrequent closed itemset are infrequent;
5. the set of maximal frequent itemsets is identical to the set of maximal frequent closed itemsets;
6. the support of a frequent itemset I which is not closed is equal to the support of the smallest frequent closed itemset containing I .

It goes through two phases. First, it generates all the frequent closed itemsets from the database. Then, since the set of all the maximal frequent itemsets is the same as the set of all the maximal frequent closed itemsets, we can derive all the frequent itemsets from this set. The more correlated the data, the more efficient the algorithm.

The closure operator that is used in this algorithm is $h = f \circ g$ (cf. Equation (4.3) and Equation (4.4)).

4.4.2 Fuzzification of the Close Algorithm

In the original Close algorithm, the closure operator deals with crisp data. In order to fuzzify it, we have to find a closure operator able to manage fuzzy data. It still takes as an argument a crisp set, which we call a *generator*, and also returns a crisp set. However, the relation \mathcal{R} between transactions and items is now fuzzy. Thus we have a fuzzy formal context.

We first define what a fuzzy set of items (respectively transactions) is. This will be used in the definition of a fuzzy closure operator.

Definition 17: Fuzzy Set of Items and Fuzzy Set of Transactions

A fuzzy set of items F_I is a fuzzy set whose membership function is defined as

$$\mu_{F_I} : \mathcal{I} \rightarrow [0; 1] \quad (4.6)$$

A fuzzy set of transactions F_T is a fuzzy set whose membership function is defined as

$$\mu_{F_T} : \mathcal{T} \rightarrow [0; 1] \quad (4.7)$$

A fuzzy closure operator verifies the same conditions as a crisp closure operator, except that it deals with fuzzy sets.

Definition 18: Fuzzy Closure Operator (Belohlávek, 2001)

Let $F_{\mathcal{U}}$ be a fuzzy set defined on a universe \mathcal{U} . A *fuzzy closure operator* h over $F_{\mathcal{U}}$ is defined as $h : F_{\mathcal{U}} \rightarrow F_{\mathcal{U}}$ and satisfies the following conditions:

- $\forall I \subseteq F_{\mathcal{U}}, I \subseteq h(I),$
- $\forall I \subseteq F_{\mathcal{U}}, h(h(I)) = h(I),$
- $\forall I, J \subseteq F_{\mathcal{U}}, I \subseteq J \Rightarrow h(I) \subseteq h(J).$

The fuzzy inclusion is defined in Appendix B on page 171.

Although this fuzzy closure operator is able to deal with a fuzzy formal context, its input argument and the result it generates are fuzzy sets. Thus, we will have to build a new closure algorithm that is based on a fuzzy closure operator and that takes and returns a crisp set.

Similarly to the crisp case, a fuzzy Galois connection enables to get a fuzzy closure operator.

Definition 19: Fuzzy Galois Connection (Belohlávek, 1999)

Let \mathcal{U} and \mathcal{W} be two universes. A *fuzzy Galois connection* between \mathcal{U} and \mathcal{W} is a pair (\uparrow, \downarrow) of mappings $\uparrow: F^{\mathcal{U}} \rightarrow F^{\mathcal{W}}$ and $\downarrow: F^{\mathcal{W}} \rightarrow F^{\mathcal{U}}$ such that

- $\forall A_1, A_2 \in F^{\mathcal{U}}, \text{Subs}(A_1, A_2) \leq \text{Subs}(A_2^{\uparrow}, A_1^{\uparrow}),$
- $\forall A \in F^{\mathcal{U}}, A \subseteq A^{\uparrow\downarrow},$
- $\forall B_1, B_2 \in F^{\mathcal{W}}, \text{Subs}(B_1, B_2) \leq \text{Subs}(B_2^{\downarrow}, B_1^{\downarrow}),$
- $\forall B \in F^{\mathcal{W}}, B \subseteq B^{\downarrow\uparrow}.$

with $F^{\mathcal{U}}$ (respectively $F^{\mathcal{W}}$) the set of all the fuzzy sets in the universe \mathcal{U} (respectively \mathcal{W}) and Subs the subsethood degree defined in Appendix B.4.5 on page 174.

Let us now introduce two new operators that form a fuzzy Galois connection when used with the Łukasiewicz implication.

Definition 20: Operators \uparrow and \downarrow (Belohlávek, 1999)

Let F_T be a fuzzy set of transactions and F_I be a fuzzy set of items. Let \rightarrow be a fuzzy implication (cf. Appendix B.4.4). Operators \uparrow and \downarrow are defined as follows

$$\forall i \in \mathcal{I}, \mu_{F_I^{\uparrow}}(i) = \inf_{t \in \mathcal{T}} (\mu_{F_T}(t) \rightarrow \mathcal{R}(t, i)) , \quad (4.8)$$

$$\forall t \in \mathcal{T}, \mu_{F_T^{\downarrow}}(t) = \inf_{i \in \mathcal{I}} (\mu_{F_I}(i) \rightarrow \mathcal{R}(t, i)) . \quad (4.9)$$

F_T^{\uparrow} is a fuzzy set of items and F_I^{\downarrow} is a fuzzy set of transactions. In the following, the composition $\uparrow \circ \downarrow$ of these two functions is written $\uparrow\downarrow$.

Definition 21: Łukasiewicz implication

The Łukasiewicz implication \xrightarrow{L} is defined as

$$\begin{aligned} \xrightarrow{L}: [0; 1] \times [0; 1] &\rightarrow [0; 1] \\ (a, b) &\mapsto \min(1 - a + b, 1) \end{aligned} \quad (4.10)$$

Using the Łukasiewicz implication enables the pair (\uparrow, \downarrow) to form a *fuzzy Galois connection* (Belohlávek, 1999). Also, this implication is compatible with the implication from classical logic. In the following, we always use \xrightarrow{L} as the implication involved in operators \uparrow and \downarrow .

Property 2: (Belohlávek, 2012; Gerla, 2013)

Let F_I be a fuzzy set of items. $\uparrow\downarrow$ is a fuzzy closure operator over F_I .

The fuzzy closure of F_I by $\uparrow\downarrow$ is $F_I^{\uparrow\downarrow}$, which is also a fuzzy set of items.

In our case, the closure operator takes a crisp set of items as a generator. Let I be a crisp set of items. It can be turned into a fuzzy set F_I to be used by the fuzzy closure operator as follows:

$$\forall i \in \mathcal{I}, \mu_{F_I}(i) = \begin{cases} 1, & \text{if } i \in I \\ 0, & \text{otherwise} \end{cases} . \quad (4.11)$$

As for the set the closure operator returns, it also has to be a crisp set. We would like to return the set of items whose membership degree to the fuzzy closure is equal to 1, such that $\forall i \in \mathcal{I}, \mu_{F_I^{\uparrow\downarrow}}(i) = 1$. This operation actually consists in computing the core (cf. Appendix B.1.2 on page 171) of $F_I^{\uparrow\downarrow}$. Thus, we propose the following closure operator:

$$\begin{aligned} h: \mathcal{P}(\mathcal{I}) &\rightarrow \mathcal{P}(\mathcal{I}) \\ I &\mapsto \text{core}(F_I^{\uparrow\downarrow}) \end{aligned} \quad (4.12)$$

with F_I the fuzzy set corresponding to I as defined in Equation (4.11). This operator is still a closure operator, but a crisp one. One can interpret the result it returns as the set of items that are shared by all the transactions that have all the items from the generator I . The proof that h is a closure operator is presented in Appendix C on page 175.

4.4.3 Description of the Algorithm

The new algorithm we propose is represented in Algorithm 1 and Algorithm 2. Its overall structure is the same as the original crisp algorithm but the computation of the closure has been modified so that it can deal with fuzzy data.

FCC_p refers to the set of triplets associated with all the Frequent Closed Candidate itemsets whose generator's size is p . FC_p refers to the set of triplet associated with all the Frequent Closed itemsets whose generator's size is p . Each triplet is represented under the form (generator, closure, support). Thus, in the following, for any $c \in FCC_p$ or FC_p , c refers to the triplet, $\text{generator}(c)$ is the corresponding generator, $h(c)$ is its closure and $\text{support}(c)$ is its support.

Algorithm 1: Fuzzy Close algorithm

```

input :
  • a fuzzy formal context  $(\mathcal{T}, \mathcal{I}, \mathcal{R})$ 
  • a minimum support  $S \in [0; 1]$ 
output: the set of all frequent closed itemsets and their support

1  $FCC_1 = \{(i, \emptyset, 0) \mid i \in \mathcal{I}\}$ 
2  $p \leftarrow 1$ 
3 while  $FCC_p \neq \emptyset$  do
4    $FCC_p \leftarrow \text{generateClosures}(FCC_p);$                                      // cf. Algorithm 2
5   forall candidate closed itemsets  $c \in FCC_p$  do
6     if  $\text{support}(c) \geq S$  then
7        $FC_p \leftarrow FC_p \cup \{c\}$ 
8     end
9   end
10   $FCC_{p+1} \leftarrow \text{generateGenerators}(FC_p)$ 
11   $p \leftarrow p + 1$ 
12 end
13  $FC \leftarrow \bigcup_{j=1}^{p-1} \{c \mid c \in FC_j\}$ 
14 return  $FC$ 

```

Algorithm 2: generateClosures function

input : the set of candidate closed itemsets FCC_p
output: updated FCC_p after the computation of closures and supports

```

1  $n \leftarrow 0$ 
2 forall  $c \in FCC_p$  do
3    $g \leftarrow \text{generator}(c)$ 
4   numbers in  $\mu_{g^{\uparrow\downarrow}}^a \leftarrow 1$ 
5    $h(c) \leftarrow \emptyset$ 
6 end
7 forall transaction  $t \in \mathcal{T}$  do
8    $n \leftarrow n + 1$ 
9   forall  $c \in FCC_p$  do
10     $g \leftarrow \text{generator}(c)$ 
11     $\mu_{g^{\downarrow}}^b \leftarrow 1$ 
12    forall items  $i \in g$  do
13       $\mu_{g^{\downarrow}} \leftarrow \min(\mu_{g^{\downarrow}}, \mathcal{R}(t, i))$ 
14    end
15    forall items  $i \in \mathcal{I}$  do
16       $\mu_{g^{\uparrow\downarrow}}(i) \leftarrow \min(\mu_{g^{\uparrow\downarrow}}(i), 1 + \mathcal{R}(t, i) - \mu_{g^{\downarrow}})$ 
17      if  $n = \text{Card}(\mathcal{T})$  then
18        if  $\mu_{g^{\uparrow\downarrow}}(i) = 1$  then
19           $h(c) \leftarrow h(c) \cup \{i\}$ 
20        end
21      end
22    end
23     $\text{support}(c) \leftarrow \text{support}(c) + \mu_{g^{\downarrow}}$ 
24  end
25 end
26 return  $FCC_i$ 

```

^a $\mu_{g^{\uparrow\downarrow}}$ is a vector corresponding to the membership function of the fuzzy closure $g^{\uparrow\downarrow}$.

^b $\mu_{g^{\downarrow}}$ is a fuzzy number that corresponds to $\mu_{g^{\downarrow}}(t)$.

Transactions	Items				
					
t_1	0.8	0.1	0.9	0.8	0
t_2	0	0.3	0.2	0	0.9
t_3	1	0.7	0.7	1	0.6
t_4	0	0.2	0	0.2	1
t_5	0.9	0.6	0.8	1	0.9

TABLE 4.4: The fuzzy database $\mathcal{D}_{\text{fuzzy}}$.

Algorithm 1 describes the process of generating all the frequent closed itemsets. On line 1, FCC_1 is initialized with every item in the set of items \mathcal{I} . On line 4, for each generator in FCC_p , the `generateClosures` function provides the corresponding closure and support. This function is detailed in Algorithm 2. Then, from line 5 to 9, the set of candidate closed itemsets FCC_p is pruned to get the set of frequent closed itemsets FC_p . New generators, whose size is $p + 1$, are generated on line 10 using the `generateGenerators` function. This function is the same as in the original algorithm (Pasquier et al., 1999). The whole process will last until no new generators can be generated. The output is the set of all frequent closed itemsets that will be used to generate all frequent itemsets.

The `generateClosures` function is stated as shown in Algorithm 2. This function has been designed to compute the closures and the supports of the generators in FCC_p performing only one database pass. For each transaction $t \in \mathcal{T}$, for each element $c \in FCC_p$, g is the generator associated to c and $\mu_{g\downarrow}(t)$ is computed looping over the items in g (from line 12 to line 14). Then, for each item $i \in \mathcal{I}$, the membership function $\mu_{g\uparrow}$ of the fuzzy closure is updated (line 16). When the last transaction is reached and there is no more update to the membership function, the core of the fuzzy closure is computed (from line 17 to line 21).

The function `generateGenerators` is exactly the same as in the Close algorithm. This function generates all the potential generators of size $p + 1$ from the generators in FC_p . In order to get one potential generator, two generators from FC_p that have the same $p - 1$ first elements are combined. Then, this set of potential generators is pruned to avoid useless computations. In particular, if one of the new generators is included in the closure of one of the former generators, then it is pruned.

Overall, the whole algorithm, i.e. Algorithm 1, needs one database pass per iteration. Like the crisp algorithm, it takes advantage of the correlation in data to have a smaller number of iterations.

After this phase, all the frequent closed itemsets are used to find all the frequent itemsets. This new phase is exactly the same as in the original Close algorithm. The first step consists in splitting the set of all frequent closed itemsets into several subsets L_p with L_p the subset of all the frequent closed itemsets of size p . Then, these new sets L_p are looped over in descending order of size to generate all frequent itemsets of size $p - 1$. The process will finish when the set of frequent 1-itemsets is completed.

4.4.3.1 Complexity Analysis

We assume in this section that the value of the minimum support is small enough so that there exist frequent itemsets.

In the best-case scenario, when all transactions are perfectly correlated, only one iteration is performed in Algorithm 1 because the closures of all 1-itemsets will be equal to \mathcal{I} . The number of operations to compute is then $O(\text{Card}(\mathcal{T}) \times \text{Card}(\mathcal{I})^2)$. Thus, the whole database is looped over just once.

In the worst-case scenario, all itemsets are both frequent and closed. Algorithm 1 would then have a complexity in $O(2^{\text{Card}(\mathcal{I})})$.

Overall, the complexity mainly depends on the number of items in the database and on the number of closed itemsets. As for the original Close algorithm, a rule of thumb for fuzzy Close is to compute the proportion of 1-itemsets that are closed. If few of them are closed, then the algorithm should perform better than other methods.

4.4.3.2 Example

For the sake of comprehension, we apply in this section the algorithm on the database $\mathcal{D}_{\text{fuzzy}}$, which is the same as in Table 4.3 and is recalled in Table 4.4. $\mathcal{D}_{\text{fuzzy}}$ contains five transactions and five items. We set the value of the minimum support at 0.4.

The pruning of FCC_1 leads to removing $\{\text{🍌}\}$ since its support is smaller than the minimum support. The other elements from FCC_1 are kept to generate FC_1 . This corresponds to line 5 to line 9 in Algorithm 1. FCC_1 and FC_1 are shown in Table 4.5.

Then, on line 10, FCC_2 is generated. $\{\text{🍌}, \text{🍇}\}$ is not a generator in FCC_2 because it is included in the closure of $\{\text{🍌}\}$. FC_2 is then generated. $\{\text{🍌}, \text{🍇}\}$ and $\{\text{🍌}, \text{🍉}\}$ have the same closure, so only one of them is kept. FCC_2 and FC_2 are shown in Table 4.6.

FC_2 contains only one element, that is why FCC_3 is empty. That is the end of the first phase, which corresponds to Algorithm 1. FC is returned. It is shown in Table 4.7.

The second phase consists in deriving frequent itemsets from frequent closed itemsets. The longest closed itemset contains three items. That is why three different sets are generated for deriving frequent itemsets: L_3 , L_2 et L_1 . Bold itemsets are itemsets which have been derived from a bigger closed itemset. These three sets are shown in Table 4.8. They contain all the frequent itemsets.

4.4.3.3 Experimental Results

Appendix D on page 179 presents experimental results on traditional data mining datasets. Fuzzy Close is compared to the fuzzy version of the Apriori algorithm (Agrawal et al., 1993) and to the best fuzzy pattern growth algorithm, UBFFPT (Lin et al., 2010a).

The results show that our algorithm achieves better performance on highly correlated datasets, which was our objective.

4.5 Discussion

In this chapter, we presented a new approach for learning relevant descriptors of classes. We made the assumption that a relevant descriptor is a frequent subset of relations, which will be tested in our experiments. That is why our approach relies on mining frequent itemsets in the training set. While the field of fuzzy frequent itemset mining is composed of several different methods, none of them uses closed itemsets to perform better on highly correlated data. Inspired by the Close algorithm, we proposed a new method that generates all the frequent itemsets in a fuzzy formal context using a closure operator. Thus, we are able to learn relevant descriptors for each class of entities.

Besides, in order to build rules for classification or constraints for annotation, we will need to extract discriminative information from these descriptors. This is tackled in Chapter 6.

Based on the learning process we just presented, we will propose in the following chapter a heuristic that aims at preventing online the evaluation of the relations that are bound to be infrequent during the evaluation process.

Generator	Closure	Support	Generator	Closure	Support
{🧀}	{🧀, 🍇}	0.54	{🧀}	{🧀, 🍇}	0.54
{🍌}	{🍌}	0.38	{🍌}	{🍌}	0.38
{🍉}	{🍉}	0.52	{🍉}	{🍉}	0.52
{🍇}	{🍇}	0.60	{🍇}	{🍇}	0.60
{🍓}	{🍓}	0.68	{🍓}	{🍓}	0.68

TABLE 4.5: FCC_1 on the left and FC_1 on the right. {🍌} is pruned from FCC_1 to FC_1 because it is not frequent.

Generator	Closure	Support	Generator	Closure	Support
{🧀, 🍉}	{🧀, 🍉, 🍇}	0.46	{🧀, 🍉}	{🧀, 🍉, 🍇}	0.46
{🧀, 🍓}	{🧀, 🍓, 🍇}	0.30			
{🍉, 🍇}	{🧀, 🍉, 🍇}	0.46			
{🍉, 🍓}	{🍉, 🍓}	0.32			
{🍇, 🍓}	{🍇, 🍓}	0.34			

TABLE 4.6: FCC_2 on the left and FC_2 on the right.

Closure	Support
{🧀, 🍇}	0.54
{🍉}	0.52
{🍇}	0.60
{🍓}	0.68
{🧀, 🍉, 🍇}	0.46

TABLE 4.7: FC

Itemset	Support	Itemset	Support	Itemset	Support
{🧀, 🍉, 🍇}	0.46	{🧀, 🍇}	0.54	{🍉}	0.52
		{🧀, 🍉}	0.46	{🍇}	0.60
		{🍉, 🍇}	0.46	{🍓}	0.68
				{🧀}	0.54

TABLE 4.8: Deriving frequent itemsets. Bold lines refer to derived itemsets. From left to right: L_3 , L_2 and L_1 .

Chapter 5

Heuristics for Preventing Redundant Evaluations

The approach we propose in this thesis requires to *evaluate fuzzy relations from a given vocabulary* before learning from them the most frequent descriptors. At the end of the evaluation step, the system has a dataset that can be represented as a formal fuzzy context on which it can perform fuzzy frequent itemset mining (cf. Chapter 4).

The time complexity of this step directly depends on the relations that are evaluated. Indeed, some relations may be *compute-intensive*, which makes the whole step longer. In order to keep it fast enough, two different types of strategy can be deployed:

- Speeding up the computation of relations based on their definition. The speed-up may be obtained by algorithm enhancement, distributed computation or approximating the result. It is a *local* optimization process (relation by relation).
- Filtering relations based on available knowledge, which is a *global* optimization process. Indeed, some evaluations of relations could be deduced from former evaluations if they satisfy some properties (symmetry, dependencies,...).

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

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

In the first section, we present the guidelines of the evaluation process. The following section tackles the heuristic that detects relations that will be infrequent for sure during the evaluation process. We then present our second heuristic. It is based on the graphical representation of the dependencies and implications between relations. This enables to get an order on relations.

5.1 Brute Force Evaluation of Relations

Evaluating the relations on entities in the instances of the training set is necessary for mining the most frequent of them.

We recall here a few notations from Chapter 3. Let $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ be a training set of n instances. Each instance $\mathbf{x}_i \in \mathcal{D}$ is divided into K entities that form the set $\mathcal{O}_{\mathbf{x}_i}$. Given a vocabulary \mathcal{V} of $n_{\mathcal{V}}$ fuzzy relations, the number of evaluations to perform for example \mathbf{x}_i is $|E_{\mathbf{x}_i}(\mathcal{V})|$, whose expression is specified by Property 1 on page 51. Therefore the total number $|E_{\mathcal{D}}(\mathcal{V})|$ of evaluations to compute on the whole training set is:

$$|E_{\mathcal{D}}(\mathcal{V})| = \sum_{i=1}^n \sum_{j=1}^{n_{\mathcal{V}}} \frac{|\mathcal{O}_{\mathbf{x}_i}|!}{(|\mathcal{O}_{\mathbf{x}_i}| - \alpha(\mathcal{R}_j))!} \quad (5.1)$$

This quantity increases fast with the number of entities in instances, the number of instances and the number of relations. This is why we worked on two heuristics that enable to prune the evaluation space.

5.2 Online Pruning of Infrequent Relations

Instances in the training set are evaluated one by one. Thus, the support of each relation can be updated each time an instance has been fully analyzed. Also, to ensure that the frequency assumption holds, **the minimum support S is always set to a value greater than or equal to at least 0.50**. We assume here that a relation that is, on average, fully satisfied in less than half of the instances in the training set is not representative of the class under study. This assumption is not restrictive since we do not expect to classify and explain an instance on the basis of such a relation. Indeed, this could harm the performance of the model and the reliability of the explanations. Thus, that enables to detect relations whose current support prevents their final support to be greater than or equal to 0.5. Such relations would be infrequent and are therefore discarded, which is similar to the construction of the frequent-pattern tree in FP-growth algorithms (Han et al., 2000) where only frequent 1-itemsets are retained. This also presents the advantage of being independent on the vocabulary and the task to perform.



Transactions	Items				
					
t_1	0.8	0.1	0.9	0.8	0
t_2	0	0.3	0.2	0	0.9
t_3	1	0.7	0.7	1	0.6
t_4	0	0.2	0	0.2	1
t_5	0.9	0.6	0.8	1	0.9

TABLE 5.1: The fuzzy database $\mathcal{D}_{\text{fuzzy}}$. For a minimum support greater than or equal to 0.5, we can know that $\{\text{carrot}\}$ cannot be frequent after having processed the first four transactions. However, for $\{\text{watermelon}\}$, we need its evaluation in t_5 to assess whether it is frequent or not.

For example, let us consider the database $\mathcal{D}_{\text{fuzzy}}$ (which is exactly the same database as in Chapter 4) given in Table 5.1. Using the same notations as in Chapter 4, $\mathcal{D}_{\text{fuzzy}}$ forms a fuzzy formal context $(\mathcal{T}, \mathcal{I}, \mathcal{R})$. With our assumption, we have $S \geq 0.5$. For an itemset $i \in \mathcal{I}$ and $k \in \llbracket 1; n \rrbracket$, let us note $S_k(i)$ the support of i after k examples. For instance, we have here $S_4(\{\text{orange}\}) = 0.325$. So, in the best case, if $\mathcal{R}(t_5, \text{orange}) = 1$, we would get $S_5(\{\text{orange}\}) = S(\{\text{orange}\}) = 0.46$. This value is lower than 0.5, so we can discard the item  after the first four transactions. However, for the itemset $\{\text{apple}\}$, we need the five transactions to assess whether it is frequent or not.

Using the same reasoning as in the previous example, we can obtain the condition on which an item can be discarded for any value of the minimum support.

Property 3

Let $i \in \mathcal{I}$ be an item. Let k be an integer in $\llbracket 1; n - 1 \rrbracket$. Let $S_k(i)$ be the support of i after k examples. Let S be the value of the minimum support. i is bound to be infrequent if, and only if, it verifies

$$S_k(i) < \frac{n(S-1)}{k} + 1 \quad (5.2)$$

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

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

$$S_n(i) = \frac{kS_k(i) + \sum_{j=k+1}^n \mathcal{R}(t_j, i)}{n}, \forall k \in \llbracket 1; n - 1 \rrbracket \quad (5.3)$$

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

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

which leads to

$$S_k(i) < \frac{n(S-1)}{k} + 1 \quad (5.4)$$

□

B only depends on S and on the size of the dataset, which makes this heuristic convenient for any task and vocabulary. With our assumption, we have $S \geq 0.5$ and thus, in the worst case when $S = 0.5$, we have $B = 1 - \frac{n}{2k}$. If S is required to be greater than 0.5, it is possible to get a higher value of B , which enables to discard even more relations. In Figure 5.1, we represented B for several values of S and for $n = 30$. Values of S that are lower than 0.50 are not considered since it would mean that either our assumption that a relevant relation is frequent is false or that the vocabulary we use is too poor or not suited to the dataset. We chose n so that its magnitude is representative of the size of the training sets we dealt with in our experiments (cf. Chapter 9).

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

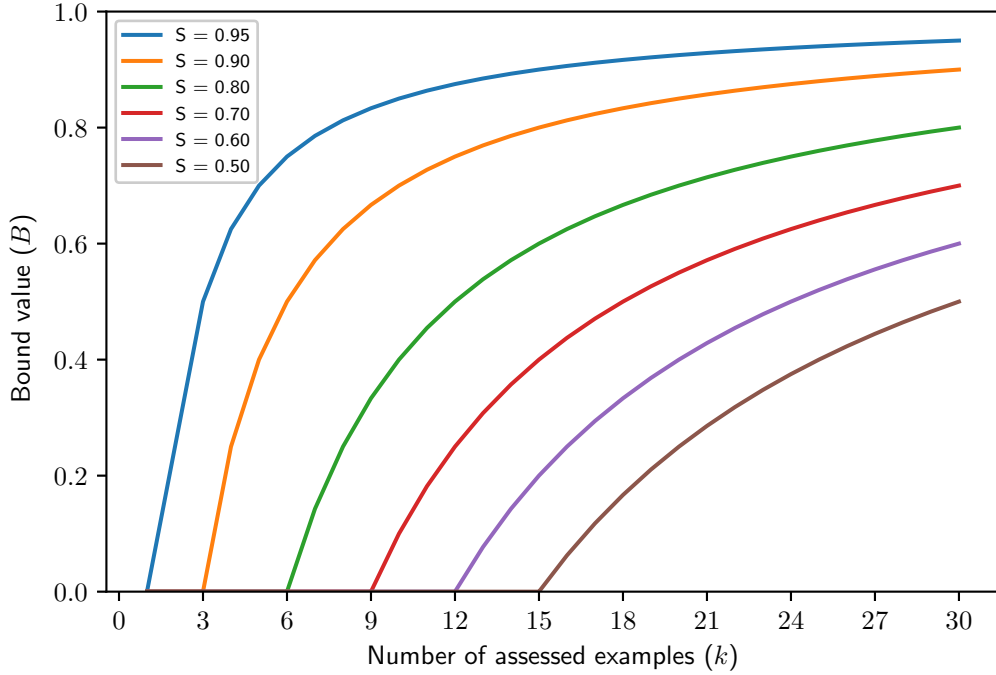


FIGURE 5.1: Curves representing the evolution of $B = \frac{n(S-1)}{k} + 1$ with the number of assessed examples k for a total number of examples $n = 30$. For the sake of clarity, only the positive values of B are displayed (if B is negative, no support can be discarded because it is always positive or null).

The experiments that we present in Chapter 9 enable us to quantify the impact of this heuristic and to validate it.

5.3 Knowledge-based Ordering of Relations

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

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

5.3.1 Dependency

The first link between relations that we present is the *dependency*.

Definition 22: Dependency

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

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

where e is a set of p entities in A .

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

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

5.3.2 Symmetry

The second type of link represents a specific property of relations: *symmetry*.

Definition 23: symmetry

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

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

For any p -ary symmetric relation \mathcal{R} , for a set \mathcal{O}_x of p entities associated to an instance x , the number of evaluations to compute involving \mathcal{R} is (according to Section 3.1)

$$|E_x(\mathcal{R})| = \frac{|\mathcal{O}_x|!}{(|\mathcal{O}_x| - p)!} . \quad (5.5)$$

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

$$|E_x(\mathcal{R})| = \frac{|\mathcal{O}_x|!}{p! (|\mathcal{O}_x| - p)!} . \quad (5.6)$$

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

5.3.3 Implications

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

We are interested in the following four kinds of implications:

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

Since the relations we use are not Boolean, we have to use fuzzy implications. In Section 4.4.2, we presented the Łukasiewicz implication that we use for generating the closure of an itemset. We use the same fuzzy implication here. To have an approach that is always consistent, we are only interested in situations where the fuzzy implication is equal to 1. The four following subsections are dedicated to establish properties for detecting relations verifying an implication. We also specify how to propagate the results for each implication.

5.3.3.1 Implication between two relations

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

$$\min(1 + b - a, 1) = 1 \Leftrightarrow 1 + b - a \geq 1 \Leftrightarrow b \geq a \quad (5.7)$$

As a consequence, we have the following property:

Property 4: $\mathcal{R}_1 \Rightarrow \mathcal{R}_2$

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

$$\forall e \in A^p, \mathcal{R}_2(e) \geq \mathcal{R}_1(e) \quad (5.8)$$

Proof. According to Equation (5.7), for each $e \in A^p$, we have

$$\xrightarrow{L} (\mathcal{R}_1(e), \mathcal{R}_2(e)) = 1 \text{ if, and only if, } \mathcal{R}_1(e) \leq \mathcal{R}_2(e). \quad \square$$

To be able to deduce the exact value of $\mathcal{R}_2(e)$ given $\mathcal{R}_1(e)$, the relation between $\mathcal{R}_1(e)$ and $\mathcal{R}_2(e)$ must be an equality. Since the upper bound on our fuzzy relations is 1, we have $\mathcal{R}_2(e) \leq 1$. If $\mathcal{R}_1(e) = 1$, then we have also $\mathcal{R}_2(e) \geq 1$ according to Property 4. That gives $\mathcal{R}_2(e) = 1$. So that implication enables to propagate the relations that are fully satisfied.

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

5.3.3.2 Implication between a relation and the complement of another relation

We consider here the implication $\mathcal{R}_1 \Rightarrow \overline{\mathcal{R}_2}$. We can perform a reasoning similar to what we did for the previous implication. We also need to introduce the notion of *complement* of a fuzzy set.

Definition 24: fuzzy complement

Let F be a fuzzy set defined on a universe \mathcal{U} and associated to the membership function μ_F . The fuzzy complement of F is defined by a membership function $\mu_{\overline{F}}$ such that $\forall u \in \mathcal{U}, \mu_{\overline{F}}(u) = c(\mu_F(u))$ with $c : [0; 1] \rightarrow [0; 1]$ a function that verifies:

- $c(0) = 1$ and $c(1) = 0$,
- $\forall z_1, z_2 \in [0; 1]$, if $z_1 < z_2$, then $c(z_1) > c(z_2)$,
- c is a continuous function,
- $\forall z \in [0; 1], c(c(z)) = z$.

In this work, we use the *standard complement*, which is defined as

$$\begin{aligned} c : [0; 1] &\rightarrow [0; 1] \\ z &\mapsto 1 - z \end{aligned} \tag{5.9}$$

We have the following property:

Property 5: $\mathcal{R}_1 \Rightarrow \overline{\mathcal{R}_2}$

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

$$\forall e \in A^p, \mathcal{R}_2(e) \leq 1 - \mathcal{R}_1(e) \tag{5.10}$$

Proof. For each $e \in A^p$, we have according to Equation (5.7)

$$\xrightarrow{L} (\mathcal{R}_1(e), \overline{\mathcal{R}_2}(e)) = 1 \text{ if, and only if, } \mathcal{R}_2(e) \leq 1 - \mathcal{R}_1(e). \quad \square$$

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

5.3.3.3 Implication between the complement of a relation and another relation

For the implication $\overline{\mathcal{R}_1} \Rightarrow \mathcal{R}_2$, we have the following property:

Property 6: $\overline{\mathcal{R}_1} \Rightarrow \mathcal{R}_2$

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

$$\forall e \in A^p, \mathcal{R}_2(e) \geq 1 - \mathcal{R}_1(e) \tag{5.11}$$

Proof. For each $e \in A^p$, we have according to Equation (5.7)

$$\xrightarrow{L} (\overline{\mathcal{R}_1}(e), \mathcal{R}_2(e)) = 1 \text{ if, and only if, } \mathcal{R}_2(e) \geq 1 - \mathcal{R}_1(e). \quad \square$$

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

5.3.3.4 Implication between the complements of two relations

For the implication $\overline{\mathcal{R}_1} \Rightarrow \overline{\mathcal{R}_2}$, we have the following property:

Property 7: $\overline{\mathcal{R}_1} \Rightarrow \overline{\mathcal{R}_2}$

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

$$\forall e \in A^p, \mathcal{R}_2(e) \leq \mathcal{R}_1(e) \quad (5.12)$$

Proof. For each $e \in A^p$, we have according to Equation (5.7)

$$\xrightarrow{L} (\overline{\mathcal{R}_1}(e), \overline{\mathcal{R}_2}(e)) = 1 \text{ if, and only if, } \mathcal{R}_2(e) \leq \mathcal{R}_1(e). \quad \square$$

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

Table 5.2 recaps the four properties that we have just established and how to propagate the results for each type of implication.

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

TABLE 5.2: This table shows the four types of implications between relations that the approach can process. Those implications enable to propagate the result of one relation to another. \mathcal{R}_1 and \mathcal{R}_2 are two p -ary fuzzy relations defined on a space A . e is a tuple of entities defined on A^p .

5.3.4 Graph Representation

The different links between relations that we use have been presented in the previous subsections. Each type of link can be represented in a graph using the edges defined in Table 5.3. This is a *labeled directed graph*.

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

TABLE 5.3: Recap of the different kinds of link we consider between relations and their notation in the graph representation. The third column specifies how the corresponding edge is represented in a graph.

\mathcal{R}_1 and \mathcal{R}_2 are two p -ary fuzzy relations defined on a space A .

Definition 25

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

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

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

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

The corresponding labeled directed graph is represented in Figure 5.2. The meaning of the edges in the graph are specified in Table 5.3.

The original goal of this strategy is to obtain an order on relations based on the way they are linked to each other. However, the labeled directed graph we generate here can be cyclic (like in Figure 5.2), which does not enable to define an order on relations.

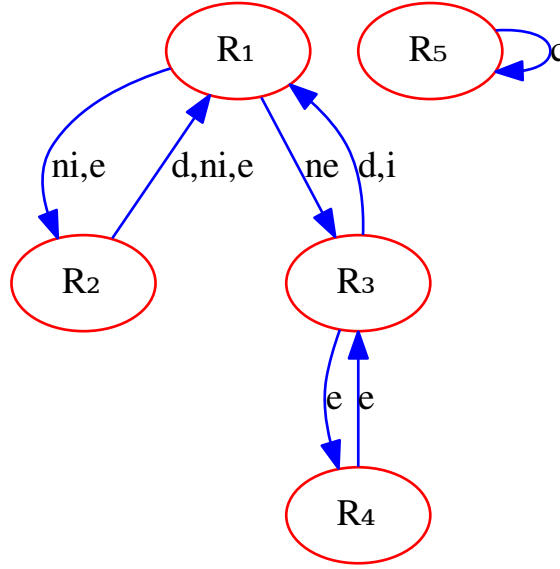


FIGURE 5.2: Example of labeled directed graph. The vertices are relations and the directed edges between vertices are labeled according to the links between relations. For each edge, its corresponding labels are to its right.

5.3.5 Knowledge-based Conversion into an Acyclic Graph

Our goal is to get an order on relations based on the knowledge graph representing them and their links. We propose to achieve that by getting rid of the cycles in the graph while keeping as much knowledge as possible to get a meaningful order. Then, once we have an acyclic graph, an order on its vertices can be obtained by topological sorting (cf. Appendix E on page 181).

The knowledge graph representing the links between relations is directed. However, in order to obtain a topological sort of this graph, it must also be *acyclic*. We introduce Algorithm 3 and Algorithm 4 that enable to obtain a directed acyclic graph from the knowledge graph we have. The idea here is to build a new acyclic graph that preserves, insofar as possible, the information contained in the original one. Thus, we can get an order on relations that is faithful to the available knowledge.

Algorithm 3 is the main algorithm for converting our labeled directed graph *LDG*, which is the knowledge graph, into a directed acyclic graph *DAG*. On line 1, *DAG* is instantiated. A copy of *LDG2* is made on line 2. The main loop starts at line 3 and loops over all the subgraphs *g* in *LDG*. At line 4, the conditional statement enables to check whether the subgraph has at least one edge or not. If the subgraph has no edge, then its vertices are added to V_{DAG} (line 5). Otherwise (from line 7), we loop over the edges of *g* (line 8). This loop first detects commutative relations and removes selfloops (line 9 to 12). Then, it extracts the dependency links and insert them in *DAG* (lines 13 to 17). We prioritize dependency over implications since the resulting relation cannot be computed without the relation it depends on. Then, the goal is to loop over *LDG* to extract implication links. The vertex used to start this exploration is selected from line 19 to line 25. We select one vertex in *DAG*, which is called *root*, that has the most outgoing edges among vertices without incoming edges (lines 20 and 21). *root* thus represents the relation involved in the most dependencies while not depending on any other relation. This is usually a generic relation so this choice enables to favour having this relation at the front of our order. If this

Algorithm 3: algorithm that converts a labeled directed graph of relations into a directed acyclic graph of relations.

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

output: a directed acyclic graph DAG

```

1  $DAG \leftarrow (V_{DAG}, E_{DAG})$  such that  $V_{DAG} = \emptyset$  and  $E_{DAG} = \emptyset$ 
2  $LDG2 \leftarrow \text{copy}(LDG)$ 
3 forall subgraph  $g = (V_g, E_g) \in G_{LDG}$  do
4   if  $E_g = \emptyset$  then
5      $V_{DAG} \leftarrow V_{DAG} \cup V_g$ 
6   end
7   else
8     forall edge  $(v_1, v_2) \in E_g$  do
9       if  $v_1 = v_2$  then
10         $V_{DAG} \leftarrow V_{DAG} \cup \{v_1^*\}^a$ 
11         $E_g \leftarrow E_g \setminus \{(v_1, v_1)\}$ 
12      end
13      else if  $d \in l(v_1, v_2)$  then
14         $V_{DAG} \leftarrow V_{DAG} \cup \{v_1, v_2\}$ 
15         $E_{DAG} \leftarrow E_{DAG} \cup \{(v_2, v_1)\}$ 
16         $E_g \leftarrow E_g \setminus \{(v_1, v_2)\}$ 
17      end
18    end
19    if  $\exists (v_1, v_2) \in V_g^2$  such that  $(v_1, v_2) \in E_{DAG}$  then
20       $\text{noParents} \leftarrow \{v \in V_{DAG} \cap V_g \mid \nexists (v_1, v_2) \in E_{DAG} \text{ such that } v_2 = v\}$ 
21       $\text{root} \leftarrow \underset{v \in \text{noParents}}{\text{argmax}} \mid \{(v, v') \in E_{DAG}\} \mid$ 
22    end
23    else
24       $\text{root} \leftarrow \text{randomly select } v \text{ in } V_g$ 
25    end
26     $\text{generateOtherLinks}(LDG, DAG, LDG2, \text{root})$ 
27  end
28 end
29 return  $DAG$ 

```

^aThe mark * is added to v_1 to specify that it represents a symmetrical relation.

Algorithm 4: algorithm that fills the directed acyclic graph with implication links. It represents the function `generateOtherLinks`.

input:

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

```

1 visited  $\leftarrow \emptyset$ 
2  $Q \leftarrow \text{queue}(\text{root})$ 
3 while  $Q$  is not empty do
4    $v_0 \leftarrow \text{dequeue}(Q)$ 
5   neighbours  $\leftarrow \{v \in V_{LDG2} \mid (v_0, v) \in E_{LDG2} \text{ or } (v, v_0) \in E_{LDG2}\}$ 
6   if  $\exists (v_1, v_2) \in E_{LDG}$  such that  $v_1 = v_0$  then
7     forall  $v_3 \in \{v \in V_{LDG} \mid (v_0, v) \in E_{LDG}\}$  do
8        $V_{DAG} \leftarrow V_{DAG} \cup \{v_3\}$ 
9        $E_{DAG} \leftarrow E_{DAG} \cup \{(v_0, v_3)\}$ 
10       $E_{LDG} \leftarrow E_{LDG} \setminus \{(v_0, v_3), (v_3, v_0)\}$ 
11    end
12  end
13  forall  $v \in \text{neighbours}$  do
14    if  $v \notin \text{visited}$  then
15      visited  $\leftarrow \text{visited} \cup \{v\}$ 
16       $Q \leftarrow \text{queue}(Q, v)$ 
17    end
18  end
19 end

```

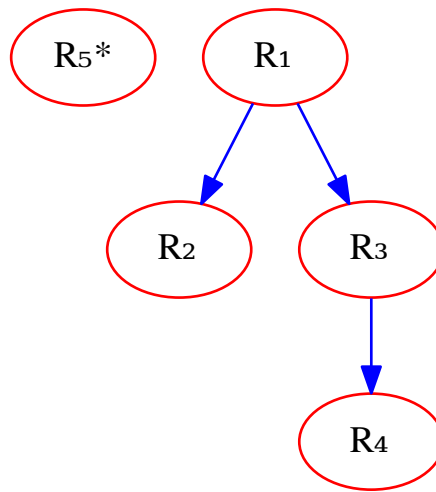


FIGURE 5.3: Example of directed acyclic graph that we get using Algorithm 3 on the knowledge graph displayed on Figure 5.2.

is not possible, we select a random vertex in V_g (line 24). Then, we call the function `generateOtherLinks(LDG, DAG, root)` that is represented in Algorithm 4 and that completes DAG . Finally, once this is over, DAG is returned.

In Algorithm 4, we loop over the graph from the vertex $root$ using a breadth-first search. We initialize a set *visited* that stores all the vertices that were visited (line 1). Then, a queue Q containing $root$ is initialized (line 2). It contains the vertices to visit. While this queue is not empty, we do the following (line 3). Q is dequeued and we call v_0 the value we get (line 4). The set of all the neighbours of v_0 is generated (line 5). Then, a loop over the outgoing edges of v_0 in G_{LDG} is performed (line 7). If there is any, we add the same edge to DAG and remove it and its opposite (if it exists) from LDG (lines 8 to 10). Finally, we update the queue and the set of visited vertices (lines 13 to 18).

Algorithm 4 is a breadth-first search algorithm, so its average complexity can be expressed as $O(\text{Card}(\mathcal{V}) + E)$ with E the number of links between relations from \mathcal{V} .

As a consequence, the average complexity of Algorithm 3 is also $O(\text{Card}(\mathcal{V}) + E)$.

Regarding the acyclicity of the new graph, there cannot be a cycle of dependencies since it would mean that no relation can be computed first, which is absurd. Also, the implications we take into account always bring two opposite edges between two relations. Indeed, if one implication entails an edge between two relations, then its contraposition will entail another edge in the opposite direction between them. Thus, since selfloops are removed (line 11 in Algorithm 3) and we ensure that parallel edges of opposite directions cannot both be added to DAG (line 10 in Algorithm 4), this new directed graph is acyclic by construction, which was the primary goal of this new method. The following example illustrates it.

Let us consider the knowledge graph represented in Figure 5.2. It is a directed graph but it is not acyclic. We apply Algorithm 3 to get a directed acyclic graph so that we can perform topological sorting. The resulting graph is represented on Figure 5.3. The star in R_5^* means that it is a symmetric relation.

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

5.3.6 Global Method

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

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

As stated in Appendix E on page 181, a graph may admit several topological sorts so the order of evaluation we get may not be unique. The following example depicts how we use the results of topological sorting to evaluate relations in a specific order.

We extracted the links between relations in the vocabulary $\mathcal{V} = \{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4, \mathcal{R}_5\}$ and represented them in a labeled directed graph in Figure 5.2. Then, Algorithm 3 converted it into a directed acyclic graph that is displayed in Figure 5.3. In this acyclic graph, there are two subgraphs corresponding to the set of vertices $\{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4\}$ and $\{\mathcal{R}_5^*\}$. After performing topological sorting, we have the following results:

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

where $\mathcal{R} \rightarrow \mathcal{R}'$ means that \mathcal{R} should be evaluated before \mathcal{R}' . Since those two topological sorts are independent on each other, we can start by evaluating either \mathcal{R}_1 or \mathcal{R}_5^* , which is a symmetric relation. That means that, for a given tuple of entities in A^p , \mathcal{R}_5^* should be evaluated only once. Let us now focus on the topological sort we got on the set of vertices $\{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4\}$. It means that \mathcal{R}_1 must be evaluated before \mathcal{R}_3 , \mathcal{R}_3 must be evaluated before \mathcal{R}_4 and \mathcal{R}_4 must be evaluated before \mathcal{R}_2 . Considering the original knowledge graph represented on Figure 5.2, this ordering of relations is justified by:

- \mathcal{R}_2 and \mathcal{R}_3 both depend on \mathcal{R}_1 , so \mathcal{R}_1 should be computed before them.
- There are two parallel edges in opposite directions between \mathcal{R}_3 and \mathcal{R}_4 on Figure 5.2. Thus, there are two possibilities and, considering only these two relations, one could be computed before the other and vice versa. The way the graph DAG is completed depends on the node that is first visited. This is the node *root*, which is defined between lines 19 and 25 in Algorithm 3. If possible, this is the node which has the most relations dependent on itself. As a consequence, in most cases, this node should not be a leaf. Here, the starting node was \mathcal{R}_1 and so \mathcal{R}_3 is visited before \mathcal{R}_4 . That is why the edge between \mathcal{R}_3 and \mathcal{R}_4 is directed toward \mathcal{R}_4 . We know that we have $\mathcal{R}_3 \Rightarrow \overline{\mathcal{R}_4}$, so if there exists $e \in A^p$ such that $\mathcal{R}_3(e) = 1$, then we get $\mathcal{R}_4(e) = 0$ without computing $\mathcal{R}_4(e)$.

5.3.7 Other applications

In this work, the heuristic we presented in the previous subsections has been applied to relation evaluation. However, it can be applied to any problem that requires having an ordering on tasks or concepts that are connected to each other. For example, it can be used for finding the optimal ordering of a sequence of tests that are dependent on each other.

5.4 Discussion

To avoid a brute force and potentially long evaluation phase, we focused in this chapter on proposing heuristics for preventing unnecessary computations of relations in the vocabulary. We proposed a first method that detects the relations that are bound to be infrequent so that they are not evaluated anymore during the training phase. The second heuristic we presented relies on the dependencies and logical implications between relations. This knowledge enables to obtain an order on relations so that evaluations are computed in an optimal order. Moreover, these two

heuristics are compatible, which enables to benefit from both of them at the same time. That leads to a shorter training phase before classifying or annotating new instances and generating explanations, which we present in Chapter 6.

Chapter 6

Generating Rules or Constraints for Performing Explainable Classification or Annotation

We have previously tackled the extraction of class descriptors and then we focused on evaluating relations in an efficient way. In this chapter, we deal with the resolution of classification and annotation problems and how to generate explanations. This is the third main step of the approach we propose (cf. Figure 2.1 on page 42). Here, we show how descriptors can be used to achieve two goals:

1. performing the target task (classification or annotation),
2. generating explanations.

Both goals are strongly related to each other since our objective is to render in a transparent way the reasoning of the model in the explanations.

For classification, we propose to build rules from the descriptors, which are frequent subsets of relations. They are transparent and can be directly constructed after the learning phase since it was performed class by class. For annotation, the problem is different because the goal is to annotate several entities in the same instance. The solution we propose is to convert descriptors into sets of constraints. Then, annotating an instance amounts to solving a fuzzy constraint satisfaction problem (FCSP).

Also, both approaches, rules for classification and FCSPs for annotation, are transparent. The way relations are combined is straightforward, which is convenient for generating an explanation. Besides, since we use fuzzy relations that are associated to a linguistic description, we can translate these combinations of relations into a natural language sentence.

The first section of this chapter tackles the building of rules for classification. Then, in the following section, we deal with the generation of constraints to define a FCSP for annotation. Finally, in the last section, we show how to generate explanations in both cases.

6.1 Building Rules for Classification

6.1.1 Classification

Classification is one of the most common task in machine learning. It consists in assigning a label to an instance.

Let \mathcal{X} be the space where the instances are defined and \mathcal{Y} be a set of l labels. Let \mathcal{D} be a dataset such as

$\mathcal{D} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n) \mid (\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}, \forall i \in \llbracket 1; n \rrbracket\}$. Instances and labels are

associated to each other by a mapping $f: \mathcal{X} \rightarrow \mathcal{Y}$. The goal of classification is to determine an estimate \hat{f} of f such as $\hat{f}: \mathcal{X} \rightarrow \mathcal{Y}$ is as close as possible to f . This is usually performed by empirical risk minimization using a loss function that assesses on a test set the difference between the results of \hat{f} and the ground truth.

However, in our work, class descriptors have been learnt by mining frequent sets of relations. So the estimate is based on these descriptors. As we saw in Chapter 4, we need both *descriptive* and *discriminative* descriptors to generate accurate explanations and perform classification. So, after the generation of the descriptors by frequent itemset mining, the estimate is based on descriptors that are modified so that they also convey discriminative information. This operation of making descriptors discriminative will be detailed in Section 6.1.3.

6.1.2 Decision Rules

In order to solve complex problems using fuzzy relations, it is necessary to combine them logically. Rules are a natural solution for that. In this subsection, we briefly present rule-based systems and then we describe how rules that involve fuzzy relations work.

6.1.2.1 Rule-based Systems

Rule-based systems are a particular case of expert systems (Bruce, 1983), which are based on the principle that rules are a natural way to represent expert knowledge (Davis et al., 1993). They offer a structuration of the knowledge involved in such approaches as a IF-THEN pair of conditions and conclusions. The principle is then to fire rules regarding the presence of facts and to observe their conclusions. In the example of classification of tangerines and oranges given in section 4.1 on page 55, we can imagine the following rule for classifying tangerines:

IF *the entity is orange AND the entity is round AND the entity is small*
THEN *the entity is a tangerine*.

These systems differ in terms of formalism used for knowledge representation (e.g. logic, fuzzy logic, etc.) and the algorithms which are used to infer new knowledge (e.g. RETE (Forgy, 1989), Mamdani inference, etc.).

With the need for explanation, everyone can observe a renewed interest in rule-based systems (Grosan and Abraham, 2011; Evans and Grefenstette, 2018): indeed, activated rules involve pieces of evidence which can be reformulated to build an explanation. For instance, the previous rule about tangerine classification could lead to the following explanation:

It is a tangerine BECAUSE it is orange, it is round and it is small.

The interpretability of this rule is due to the predicates it handles, which are understandable, and to the fact that it is not too complex. As we wrote in Chapter 1, if the model is too complex (too many rules, rules are too long), it is not considered interpretable since a human would struggle to get a global understanding of what the model does.

In our case, we chose to use fuzzy relations because they can manage the inherent vagueness of natural language better than other formalisms. Thus, they facilitate the generation of human-readable explanations.

6.1.2.2 Fuzzy Relational Rules

The rules that are used in our system involve fuzzy relations. They are called *fuzzy relational rules* (Yager, 1991) and are defined as follows.

Definition 26: Fuzzy Relational Rule (Yager, 1991; Yager and Filev, 1996)

A fuzzy relational rule is a fuzzy rule that contains a fuzzy relation in its antecedents, such as:

$$\begin{aligned} \text{IF } Z_1 \text{ is } F_1 \wedge \cdots \wedge Z_q \text{ is } F_q \wedge \left\{ \bigwedge_{i,j,k} ((Z_i, \dots, Z_j) \text{ is } \mathcal{R}_k) \right\} \\ \text{THEN } C \text{ is } F_C \end{aligned}$$

where Z_1, \dots, Z_q are the antecedent variables defined on the universes $\mathcal{U}_1, \dots, \mathcal{U}_m$, C is the consequent variable defined on the universe \mathcal{U}_C , F_1, \dots, F_q, F_C are fuzzy subsets and for each k , \mathcal{R}_k is a p -ary fuzzy relation defined on $\mathcal{U}_i \times \cdots \times \mathcal{U}_j$. \wedge is a t-norm (cf. Appendix B.4.1 on page 172).

For instance, in the previous subsection, the rule given as example to classify tangerines can be seen as a fuzzy relational rule with *unary* relations. Let us now give another rule with two different entities, such as

IF *entity1 is orange AND entity2 is orange AND entity1 is smaller than entity2*
THEN *entity1 is a tangerine*.

This is another example of fuzzy relational rule with a dyadic relation (*is smaller than*).

While the evaluation of the rule is complex in the general case, it is simple when inputs are just singletons, such as entities in our instances. For such inputs (z_1^*, \dots, z_m^*) , the activation (or strength) of the rule is (Yager and Filev, 1996)

$$A(c) = F_1(z_1^*) \wedge \cdots \wedge F_q(z_q^*) \wedge \left\{ \bigwedge_{i,j,k} \mathcal{R}_k(z_i^*, \dots, z_j^*) \right\} \wedge F_C(c), \forall c \in \mathcal{U}_C \quad (6.1)$$

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

6.1.3 Constructing Relational Rules

As explained in Chapter 4, we extract class descriptors as frequent sets of relations. The goal is now to build rules from these sets. However, in the fuzzy Close algorithm, we derive frequent itemsets from the frequent closed itemsets. As a consequence, there are many redundancies between frequent itemsets, such as $\{\text{🍌}, \text{🍉}, \text{🍇}\}$, $\{\text{🍌}, \text{🍉}\}$ and $\{\text{🍌}, \text{🍇}\}$ in Table 4.8. We can get rid of those redundancies resorting to maximal itemsets, which are defined in Definition 11 on page 58. We also know that the set of maximal frequent itemsets is the same as the set of maximal frequent closed itemsets (Pasquier et al., 1999). So we just need to find the maximal frequent closed itemsets among the frequent closed itemsets, which are given by FC at the end of Algorithm 1. This is done as described in Algorithm 5. The algorithm first sort frequent closed itemsets by descending order of size. Then, it extracts the itemsets that are not subsets of any maximal frequent closed itemset. At the end, we obtain the set of maximal frequent closed itemsets MFC . The complexity of this algorithm is equivalent to the complexity of the sort algorithm used at line 2. Thus, the average complexity is $O(N \log N)$ with N the number of frequent closed itemsets in FC .

Algorithm 5: Algorithm extracting maximal frequent closed itemsets from the set of frequent closed itemsets.

input : the set of frequent closed itemsets FC
output: the set of maximal frequent closed itemsets MFC

```

1  $MFC = \emptyset$ 
2  $FC \leftarrow \text{sort}(FC)$  by descending order of size
3 forall  $I \in FC$  do
4   if  $\nexists J \in MFC$  such that  $I \subset J$  then
5      $MFC \leftarrow MFC \cup \{I\}$ 
6   end
7 end
8 return  $MFC$ 

```

Closure	Support
 	0.54
	0.52
	0.60
	0.68
  	0.46

TABLE 6.1: The set FC of frequent closed itemsets that we got in the example of Chapter 4 on page 55.

For instance, we recall in Table 6.1 the set FC that we got in the example given in Chapter 4. Here, the maximal frequent closed itemsets, and thus the maximal frequent itemsets, are $\{\text{strawberry}\}$ and $\{\text{cheese}, \text{watermelon}, \text{grapes}\}$ so $MFC = \{\{\text{strawberry}\}, \{\text{cheese}, \text{watermelon}, \text{grapes}\}\}$.

Since the algorithm is performed class by class, rules are class-specific. Let $y \in \mathcal{Y}$ be a label corresponding to one class of entities. Let MFC_y be the set of descriptors corresponding to this class. In order to have discriminative descriptors, we remove from the descriptors all the relations that are common to several classes to get the set of discriminative descriptors MFC_y^* :

$$MFC_y^* = \left\{ I^* \mid I^* = I \setminus \bigcup_{y' \in \mathcal{Y} \setminus \{y\}} \left(\bigcup_{J \in MFC_{y'}} J \right), I \in MFC_y \right\} \quad (6.2)$$

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

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

$$\text{IF } \bigwedge_{\mathcal{R} \in I^*} \mathcal{R} \text{ THEN class} = y \quad (6.3)$$

Since there may be several descriptors for a given class, rules are actually aggregated:

$$\forall y \in \mathcal{Y}, \mathbf{x} \in \mathcal{X}, \mu_y(\mathbf{x}) = \bigvee_{I^* \in MFC_y^*} \left(\bigwedge_{\mathcal{R} \in I^*} \mathcal{R} \right) \quad (6.4)$$

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

However, this does not use any information about the support of the descriptors. Indeed, descriptors do not all have the same support and so the rules they entail should not all have the same weight in the final decision. Thus, we propose the following:

$$\forall y \in \mathcal{Y}, \mathbf{x} \in \mathcal{X}, \mu_y(\mathbf{x}) = \bigvee_{I^* \in MFC_y^*} (\text{support}(I^*) \times \bigwedge_{\mathcal{R} \in I^*} \mathcal{R}) \quad (6.5)$$

where we can interpret $\mu_y(\mathbf{x})$ as the *confidence* in assigning the label y to \mathbf{x} . Then, the label corresponding to the greatest confidence is assigned to the instance under study.

In the example presented in Table 6.1, assuming that the dataset represents instances of a class associated to label y , we saw that $MFC_y = \left\{ \{\text{🍎}\}, \{\text{🧀}, \text{🍌}, \text{🍇}\} \right\}$. Since $\{\text{🍎}\} \cap \{\text{🧀}, \text{🍌}, \text{🍇}\} = \emptyset$, we have $MFC_y^* = MFC_y$. Finally, we get

$$\mu_y(t) = \bigvee \left\{ 0.68 \times \mathcal{R}(t, \text{🍎}), 0.46 \times \bigwedge_{i \in \{\text{🧀}, \text{🍌}, \text{🍇}\}} \mathcal{R}(t, i) \right\} \quad (6.6)$$

for a transaction $t \in \mathcal{T}$ (cf. the fuzzy formal context $(\mathcal{T}, \mathcal{I}, \mathcal{R})$ presented in Table 4.3 on page 61).

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

6.2 Converting Relations into Constraints for Annotation

6.2.1 Annotation

Annotation consists in assigning a label to one or several entities in an instance. It can be seen as a specific case of classification.

We define \mathcal{X} and \mathcal{Y} the same way as for classification. As defined in Chapter 3, we define $\mathcal{O}_x = \{o_{x,1}, \dots, o_{x,K} \mid o_{x,i} \in \mathcal{X}, \forall i \in \llbracket 1; K \rrbracket\}$ as a set of K entities in \mathbf{x} that are defined on \mathcal{X} . Let $Y_x = \{y_{x,1}, \dots, y_{x,K} \mid y_{x,i} \in \mathcal{Y}, \forall i \in \llbracket 1; K \rrbracket\}$ be a set of K labels. Let $D = \{(\mathbf{x}_1, \mathcal{O}_{x_1}, Y_{x_1}), \dots, (\mathbf{x}_n, \mathcal{O}_{x_n}, Y_{x_n})\}$ be a dataset of n instances. Entities in instances and labels are associated to each other by a mapping $f: \mathcal{X}^K \rightarrow \mathcal{Y}^K$. The goal of annotation is to find a mapping $\hat{f}: \mathcal{X}^K \rightarrow \mathcal{Y}^K$ that is as close as possible to f .

In this work, we propose to convert the relations into constraints to solve a *fuzzy constraint satisfaction problem* (Dubois et al., 1996). The goal is to find the mapping between entities and labels that best satisfies the constraints, as proposed by (Vanegas et al., 2016).

6.2.2 Fuzzy Constraint Satisfaction Problem

6.2.2.1 Definitions

A constraint satisfaction problem (CSP) (Montanari, 1974; Mackworth, 1977; Waltz, 1972) consists in assigning some values to a set of variables that must respect a set of constraints. Many problems can be represented as a CSP like scheduling problems or the Golomb Ruler problem for placing sensors.

(Dubois et al., 1996) presented an extension of CSP to the fuzzy logic framework to deal with imprecise parameters and flexible constraints. This is called a fuzzy constraint satisfaction problem (FCSP).

Definition 27: Fuzzy Constraint Satisfaction Problem (Dubois et al., 1996)

A fuzzy constraint satisfaction problem is defined by:

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

In particular, if the constraints involve relations that are all unary or binary, the FCSP is called a fuzzy constraint network (Dubois et al., 1996).

One instantiation of a FCSP is evaluated by its degree of consistency.

Definition 28: Degree of Consistency (Vanegas et al., 2016)

Given an instantiation γ , its degree of consistency is:

$$\text{cons}(\gamma) = \min_{c_i \in C} \mu_{\mathcal{R}_i}(\gamma|_{V_i}) \quad (6.7)$$

where $\gamma|_{V_i}$ is the projection of γ on V_i and $\mu_{\mathcal{R}_i}$ the membership function representing \mathcal{R}_i .

This consistency degree also enables to compare different solutions so that the best one can be extracted. In the following, we write that an instantiation is inconsistent if its consistency is equal to 0. Otherwise, we say that it is consistent.

Finding consistent solutions is usually too complex, which is why it is often more convenient to use only local consistency to have a simpler problem to solve. Local consistency is formalized by the concept of k -consistency.

Definition 29: k -consistency (Dubois et al., 1996)

A FCSP is said to be k -consistent if it verifies the two following properties:

1. any consistent instantiation γ of $k - 1$ variables can be extended to a consistent instantiation γ' involving any k -th variable,
2. this instantiation γ' must be as consistent as γ .

The most used consistencies are *arc-consistency* (or 2-consistency) and *path-consistency* (or 3-consistency). In this work, we rely on arc-consistency to solve FCSPs. In the context of a FCSP, it can be defined as follows.

Definition 30: Arc-consistency (Vanegas et al., 2016)

A FCSP is *arc-consistent* if, and only if,

$$\forall (v_i, v_j) \in V^2 \text{ such as } v_i \neq v_j, \forall c_k \text{ involving } v_i \text{ and } v_j, \quad (6.8)$$

$$\forall u \in D_i, \mu_{v_i}(u) \leq \sup_{w \in D_j} \left[\min(\mu_{R_k}(u, w), \mu_{v_j}(w)) \right]$$

where $\mu_{v_i}(u)$ is the degree of membership of the variable v_i to the domain u .

In the next paragraphs, we are going to present how to use and implement arc-consistency to solve a FCSP.

6.2.2.2 Resolution

The resolution of a FCSP can be based on *backtracking*. This approach consists in incrementally building a candidate solution. As soon as a candidate solution is not consistent anymore, it goes backward, to the previous iteration, and explores other possible candidate solutions. This is done until all solutions are computed.

Each time a candidate solution is extended, its arc-consistency is checked in order to update the search space and prune inconsistent solutions. It is achieved using the FAC-3 algorithm (Dubois et al., 1996), which is described in the following paragraph.

6.2.2.3 Algorithms

A famous filtering algorithm based on arc-consistency is AC3 (Mackworth, 1977). It has been extended to FCSPs as FAC-3 in (Dubois et al., 1996). (Vanegas et al., 2016)

Algorithm 6: FAC-3 algorithm (Dubois et al., 1996)

input : FCSP (V, D, C)
output: an upper bound of the consistency degree if the problem is arc-consistent

```

1 ConsSup  $\leftarrow$  1
2 ToCheck  $\leftarrow$  C
3 while ToCheck  $\neq \emptyset$  do
4   ToCheck  $\leftarrow$  ToCheck  $\setminus \{c_i\}$ 
5   foreach  $v_{i_k} \in V_i$  do
6     Changed[ $i_k$ ]  $\leftarrow$  False
7   end
8   result  $\leftarrow$  Revise( $c_i$ , Changed, ConsSup)
9   if result = EmptyDomain then
10    return Failure
11  end
12  if result = Changed then
13    foreach  $c_l \neq c_i$  such that  $\exists v_j \in V_i \cap V_l$  and Changed[ $j$ ] = True do
14      ToCheck  $\leftarrow$  ToCheck  $\cup \{c_l\}$ 
15    end
16  end
17 end
18 return ConsSup

```

presents an updated version of the algorithm that can deal with groups of entities. It is presented in Algorithm 6. Given a set of constraints C , this algorithm propagates the constraints to update the degrees of membership of each variable v_i to its possible values in D_i . The list *ToCheck* stores the constraints to propagate and the variable *result* indicates if any update occurred (*result* = *Changed*), no update occurred (*result* = *NoChange*) or a variable does not have any possible domain anymore (*result* = *EmptyDomain*). The potential updates are performed in the method *Revise*, which is detailed in Algorithm 7. *Revise* updates the membership functions associated to each variable. As we mentioned for the variable *result*, three situations may occur once a membership function has been set to its new value (*newDegree*). Either the new value is equal to 0 and then the domain of the variable under consideration may be empty, or the new value is lower than the current one and the update is performed, or the new value is greater than or equal to the current one and nothing changes.

The FAC-3 algorithm enables to prune the search space given the current instantiation that is being built. Thus, it is called at each iteration of the backtracking, which will then explore every possible solution.

Algorithm 7: Revise method (Dubois et al., 1996; Vanegas et al., 2016)

input : c_k , Changed, ConsSup
output: *result*, which indicates if the variable memberships to domains have been updated or if a variable cannot be assigned any value

```

1 Height  $\leftarrow$  0
2 result  $\leftarrow$  NoChange
3 foreach  $v_i \in V_k$  do
4   foreach  $u \in D_i$  do
5     newDegree  $\leftarrow$  0
6     foreach tuple  $Q$  in the domain of  $\mathcal{R}_k$  such that  $u = Q|_{D_i}$  do
7       eval  $\leftarrow$   $\min(\mu_{\mathcal{R}_k}(Q), \min_{\substack{j \in \llbracket 1, K \rrbracket \\ j \neq i}} \mu_{v_j}(Q|_{D_j}))$ 
8       Height  $\leftarrow$   $\max(\text{eval}, \text{Height})$ 
9       newDegree  $\leftarrow$   $\max(\text{eval}, \text{newDegree})$ 
10    end
11    if newDegree = 0 then
12      Delete  $u$  from  $D_i$ 
13      if  $D_i = \emptyset$  then
14        return EmptyDomain
15      end
16    end
17    if newDegree <  $\mu_{v_i}(u)$  then
18      Changed[ $i$ ]  $\leftarrow$  True
19       $\mu_{v_i}(u) \leftarrow$  newDegree
20      result  $\leftarrow$  Changed
21    end
22  end
23 end
24 ConsSup  $\leftarrow$   $\min(\text{ConsSup}, \text{Height})$ 
25 return result

```

6.2.3 Generating constraints

As defined in the previous subsection, a constraint c is a pair (\mathcal{R}, V) composed of a sequence V of variables, representing entities, and one relation \mathcal{R} linking those variables. Since items represent relations that were evaluated between entities, they can be directly translated into constraints.

The generation of a set of fuzzy constraints C for defining a FCSP is analogous to the generation of rules: it is performed class by class and it is based on extracting the set of maximal frequent closed itemsets using Algorithm 5. However, unlike rules, we do not prune the itemsets from the relations that are common to several classes. The idea is that we do not want to have too few constraints in our FCSP and that we should just ensure that there is no descriptor that describe several classes. The risk if too few constraints are learnt is that there may be too many highly consistent solutions. On the other hand, we could get no consistent solution if too many constraints are learnt, but this case should not happen if the various minimum supports (for each class) are set properly.

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

$$I_y^M = \operatorname{argmax}_{I \in \mathcal{J}} [\operatorname{support}(I)] \text{ such as } \mathcal{J} = \{J \in MFC_y \mid |J| = \max_{P \in MFC_y} |P|\} \quad (6.9)$$

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

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

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

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

6.3 Generating Explanations from Rules and Constraints

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

now on the generation of explanations from the models we built for classification or annotation. The explanation we would like to provide to the end-user is a sentence in natural language that explains how a result has been produced.

This section aims at describing the process of explanation generation in this work. The first subsection presents the tool we used, SimpleNLG (Gatt and Reiter, 2009), for generating syntactically-correct sentences. The following section explains how this tool was integrated to our approach for generating explanations in natural language.

6.3.1 SimpleNLG: a Tool for Generating Sentences

In linguistics, a *realization* consists in generating a *surface form*, which is a correct sentence in a given natural language, from a more abstract representation in which the different components such as the subject or the verb are specified. Therefore, a *surface realizer* is a system that is able to take an abstract semantic representation as an input to generate a syntactically-correct sentence.

The two most famous open-source realisation engines are SimpleNLG (Gatt and Reiter, 2009) and OpenCCG (OpenCCG 2004). In this thesis, we decided to rely on SimpleNLG because it provides an API that is easier to use than OpenCCG while being complete enough for the kind of explanation we would like to generate.

It works as follows. An empty sentence s , which is actually called a *clause* in the API, is created using the function `createClause()`. s is an object that can be completed using the following functions:

- the function `setSubject(subj)` enables to specify the subject in s ,
- the function `setVerb(verb)` enables to specify the verb in s ,
- the function `setObject(comp)` enables to specify the complement in s .

For example, we can call `s.setSubject("The tangerine")`, `s.setVerb("be")` and `s.setObject("orange")` to specify the semantic representation we want. This semantic representation could also be created calling `createClause("The tangerine", "be", "orange")`. Then, calling `realiseSentence(s)` will generate: *The tangerine is orange*.

We can also generate a conjunction. Let us consider the three following sentences:

- $s_1 = \text{createClause}(\text{"The tangerine"}, \text{"be"}, \text{"orange"})$,
- $s_2 = \text{createClause}(\text{"The banana"}, \text{"be"}, \text{"yellow"})$,
- $s_3 = \text{createClause}(\text{"The cherry"}, \text{"be"}, \text{"red"})$.

Calling `createCoordinatedPhrase()` creates an empty clause expressing a conjunction. We call it c . The members of c can be specified calling `c.addCoordinate(s1)`, `c.addCoordinate(s2)` and `c.addCoordinate(s3)`. Calling `realiseSentence(c)` then generates the following sentence: *The tangerine is orange, the banana is yellow and the cherry is red*.

The final construction we need is the subordinate clause. Let us consider the two following clauses:

- $s_4 = \text{createClause}(\text{"The entity"}, \text{"be"}, \text{"a tangerine"})$,
- $s_5 = \text{createClause}(\text{"it"}, \text{"be"}, \text{"orange"})$.

Calling `s5.setFeature(Feature.COMPLEMENTISER, 'because')` and `s4.addComplement(s5)` enables to generate: *The entity is a tangerine because it is orange.*

SimpleNLG provides more functionalities to generate other types of conjunctions, subordinate clauses or manage tenses but we do not need them for the explanations we would like to produce in this thesis. In the following subsection, we present how to use the functions we just presented for generating explanations.

6.3.2 Generating Explanations in Natural Language

All the relations in the vocabulary are associated to a linguistic variable (cf. Chapter 3). That enables to express these relations in natural language. Also, the evaluated relations that have been learnt to form class descriptors involve entities whose class is known. Thus, for each relation, we can generate a description in natural language. For example, let us consider a descriptor of a class of entities β containing the relation $\mathcal{R}_{\text{equal}}(\beta, \eta)$, η being another class of entities. It can be expressed in natural language such as “ β equals η ”. Since each relation has a known and constant arity, automating the generation of an expression in natural language can be performed by associating a template to each relation. This template will be used in SimpleNLG, which will realize the corresponding clause. For instance, the template associated to $\mathcal{R}_{\text{equal}}$ is `createClause(entity1, ‘equal’, entity2)` where `entity1` is the subject, ‘equal’ is the verb and `entity2` is the object. Also, since it is straightforward to convert a constraint into a relation (cf. the definition of a constraint in Section 6.2.2), constraints can be used to generate a natural language expression as we do with relations.

At a higher level, descriptors are used to build rules (classification) or constraints (annotation). For rules, their antecedent is a conjunction of relations. Thus, all the natural language expressions should be linked by conjunctions, which can be done with SimpleNLG as shown in the previous subsection. In the case of annotation, the treatment is similar. Since the result of a FCSP is the solution that best satisfies all the constraints, it can be translated in natural language as a conjunction using the same method as for rules.

For classification, the result is a class. It is returned by an aggregation of rules (cf. Section 6.1.3). When the aggregation operator is the supremum, only one rule computes the final result and can be directly translated into a sentence in natural language using the subject-verb-complement structure. When the aggregation operator is the mean, the basis for the explanation is the union of the antecedents of all the rules that are being aggregated. We can then also generate a sentence in natural language. Let ω be the result of the classification for a given instance. The output can be generated from the following template: `createClause(‘This instance’, ‘belong’, ‘class’ + ω)`. The confidence of the system in the output it returns can also be expressed as a linguistic variable, which enables to characterize it as low, medium or high for example. With SimpleNLG, the confidence can be added to the end of the clause as an additional complement using `addComplement(‘with a’ + level + ‘confidence’)` where `level` is the linguistic expression that characterizes the confidence. The process is similar with constraints.

Finally, both parts of the explanation can be linked with a complementizer such as *because* or *since*. We then get a complete explanation in natural language. For example, classifying an instance as a tangerine could lead to the following explanation:

This instance belongs to class “tangerine” with a high confidence because an entity is orange, it is round and it is small.

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

Further works have been proposed for generating more complex explanations in natural language that can also express more logical combinations like disjunctions or negations, as in (Baaj and Poli, 2019).

6.4 Discussion

The goal of this chapter is to propose an approach that can perform classification or annotation and that is able to provide an explanation to the results it returns.

In order to get the most representative class descriptors, we retain only the descriptors corresponding to a maximal frequent closed itemset. For classification, that enables to build one fuzzy relational rule per descriptor. Then, the rule is aggregated if needed and classification can be performed. In the context of annotation, we propose to generate constraints from the largest maximal frequent itemset in order to define a fuzzy constraint satisfaction problem. The most consistent solution to this FCSP is used for annotating entities.

The generation of explanations is based on what the model does (rules or FCSP). The fuzzy relations in the vocabulary are all associated to a linguistic variable, which is convenient for translating them into natural language. The antecedents of the rules or the constraints that led to the result are seen as a conjunction of such relations. Also, we can express a level of confidence in the result which is part of the explanation. All this information is finally sent to a surface realizer to generate sentences in natural language.

As we saw in the description of the explanation generation process, our approach expresses conjunctions between relations to explain results. That is the direct consequence of the models we use. A natural evolution would lead to expressing also disjunctions or negations for example. This would imply a different learning strategy, but also another way of assessing the consistency of an instantiation in FCSPs.

Conclusion

In this part, we proposed an approach for building a model able to perform explainable classification or annotation. The primary step was to ensure that the model will be expressive enough to solve the problem. That is why the system is provided a vocabulary of potentially relevant relations. Moreover, we chose to use fuzzy relations because they are convenient for managing imprecision in data and they are well-suited for building an explainable system since they link quantitative and linguistic concepts. These relations are evaluated on the entities of the instances in the training set.

Then, the goal was to construct relevant descriptors for each class of entity. Relations that are consistently satisfied by examples of one class, and thus descriptive of this class, can be extracted mining frequent itemsets in the database of fuzzy relations that were evaluated. So descriptors are built as frequent sets of relations. Since relations are fuzzy and multiple instances from a given class should all share a common set of relations, we worked on a new fuzzy frequent itemset mining algorithm that is suited to this situation. Inspired by an algorithm that was made for dealing with non-fuzzy data, it relies on a new closure operator that takes and returns crisp sets of relations while being able to manage fuzzy formal contexts.

Knowing how class descriptors are extracted, we then proposed a heuristic that discards during the evaluation phase the relations that are bound to be infrequent. The other strategy we proposed takes advantage of the knowledge on the relations in the vocabulary. This knowledge is represented in a directed graph that links relations that are tied by a logical implication, a dependency or a symmetry. To obtain an order on relations that enables to propagate information effectively, we proposed a new algorithm for converting such a graph into a directed acyclic graph. Then, topological sorting can be applied to get the intended order so that redundant computations are avoided during the evaluation process.

Finally, we focused on how the model should use the relation-based class descriptors that were previously extracted. These descriptors can be turned into discriminative structures by removing the relations common to descriptors of other classes. In order to have a transparent model, we decided to build fuzzy relational rules relying on these descriptors for classification. In the case of annotation, we propose to generate constraints from the descriptors so that a fuzzy constraint satisfaction problem can be defined and solved using a backtracking approach and the FAC-3 algorithm. These two types of models being transparent, the way they handle relations can be translated into a natural language sentence using a surface realizer. This also relies on the fact that fuzzy relations can easily be interpreted in understandable terms. We can thus produce an explanation that expresses the reasoning of the system.

At this point of the thesis, we have a global approach that is able to perform explainable classification or annotation. We are now going to deploy it for classifying or annotating images and time series. This requires to define specific fuzzy relations for performing spatial or temporal reasoning.

Part III

Application to Spatial and Temporal Reasoning

Introduction

In Part II, we presented the theoretical foundations of our approach. After setting a vocabulary of potentially relevant relations, the first step consists in evaluating these relations on the entities in the instances of the training set. We proposed two heuristics for avoiding unnecessary computations and making the evaluation process faster. Then, frequent class descriptor are extracted using a new fuzzy frequent itemset mining algorithm. Finally, discriminative rules or constraints can be generated and used to make a decision. An explanation, based on those rules/constraints, can then be generated.

In order to show the genericity of our approach, we are going to tackle several situations involving spatial or temporal information. The only difference between these various situations will be the vocabulary of relations that needs to be set beforehand.

Setting a vocabulary is an important step because it conditions the remaining of the approach. In Chapter 7, we present the most important spatial and temporal fuzzy relations in the literature that could integrate a vocabulary. These relations all express human-understandable concepts that can be used for generating an explanation in natural language. Also, we will briefly introduce spatio-temporal fuzzy relations since they rely on the notions presented for spatial and temporal relations.

Among the spatial relations we presented, directional ones have received much attention from the beginning of the modeling of spatial relations (Freeman, 1975). As an essential family of relations, they are not only important in many visual tasks but can also be used to compute more complex fuzzy relations, such as *between* (Cinbis and Aksoy, 2007) or *parallelism* (Vanegas et al., 2012). A common approach to compute fuzzy directional relations consists in computing a *fuzzy dilation* (Bloch, 1999a). This operation being expensive, a few approximation method were proposed (Bloch, 1999a; Wang et al., 2006) but they either do not make the most of modern parallel architectures or are not suited to deal with various image sizes. In Chapter 8, we present a new algorithm that avoids unnecessary computations (compared to the original computationally expensive algorithm) and we propose highly parallel implementations that can take advantage of modern CPU capabilities. We show that these implementations are fast and compute the exact value of the fuzzy dilation.

In Chapter 9 and Chapter 10, we present the experiments we carried out to assess our approach. Chapter 9 focuses on the applications to images. A first experiment was conducted on a toy dataset of images to perform classification. It aims at presenting the behaviour of the model we built and the explanations it provides. We then tackled organ annotation in 2D medical images to evaluate how the model performs in real conditions given the segments corresponding to each organ. In that experiment, we assessed more accurately the explanations produced by the model by organizing a survey. In Chapter 10, we present a case of time-series classification of toxic chemicals. In particular, we evaluated how the model behaves when entities are segmented in an unsupervised way and how it deals with a simpler vocabulary.

Chapter 7

Spatial and Temporal Relations

As we presented in Part II, the approach we propose relies on learning the most relevant relations for representing a class of entities. These relations are provided in a vocabulary, which is a set of relations. It has a direct impact on the expressivity of the system, which is why the vocabulary should be generated wisely. Moreover, we decided to focus on fuzzy relations since they enable to manage vagueness and can be associated to a linguistic variable. Indeed, this is convenient for building an XAI since it enables to handle natural language expressions (Zadeh, 1999).

In this chapter, we are going to present fuzzy relations that could be relevant for the problems we will tackle in Chapter 9. Thus, these relations are candidates for building a vocabulary. We focused on the two following types of fuzzy relations:

- Fuzzy *spatial* relations: they are important for dealing with images since the spatial arrangements of entities is very important in scene understanding (Biederman, 1981). Moreover, the fuzzy set theory is convenient for expressing spatial relations (Freeman, 1975).
- Fuzzy *temporal* relations: they enable to detect local properties or patterns in time series, which is important for solving many time series classification problems (Geurts, 2001).

Also, as a consequence of the two previous points, there also exist fuzzy *spatio-temporal* relations for dealing with sequences of images or geolocated entities. We will present them briefly.

We first lay emphasis on fuzzy spatial relations in Section 7.1. They can be split into three categories: topological, metric and structural relations (Vanegas et al., 2016; Hudelot et al., 2008). Then, in Section 7.2, we present a review of fuzzy temporal relations. Finally, we will also briefly tackle fuzzy spatio-temporal relations in Section 7.3.

7.1 Fuzzy Spatial Relations

The coherence of the spatial arrangement of entities is very important in scene understanding (Biederman, 1981; Bloch, 2005). That is why we mainly concentrate on spatial relations when dealing with images. An extensive review of this type of relations is given in (Bloch, 2005).

Spatial relations belong to one of the three following categories: topological, metric and structural relations. The three following subsections are dedicated to each one of these categories.

Images are defined in a universe, or space, composed of pixels. We assume here that an entity is a region in an image that can be represented as a fuzzy set. It enables to deal with entities whose borders are not well known.

7.1.1 Topological Relations

Topological relations enable to express core spatial configurations such as the adjacency between two regions or their overlapping. In order to model such relations, (Randell et al., 1992) proposed the Region Connection Calculus (RCC). In particular, RCC8 is its most famous version. It consists in eight basic binary relations and it relies on a reflexive and symmetric relation C that models the connection between two regions. These relations are displayed in Figure 7.1 and defined in Table 7.1. Those are crisp relations between crisp regions.

In (Schockaert et al., 2008b), the authors propose a fuzzy region connection calculus. The core relations are fuzzy relations that represent the same concepts as RCC8 relations. Regions can be fuzzy or not. C is a reflexive and symmetric fuzzy relation modeling the connection between two regions. The definitions of all these fuzzy relations are shown in Table 7.1.

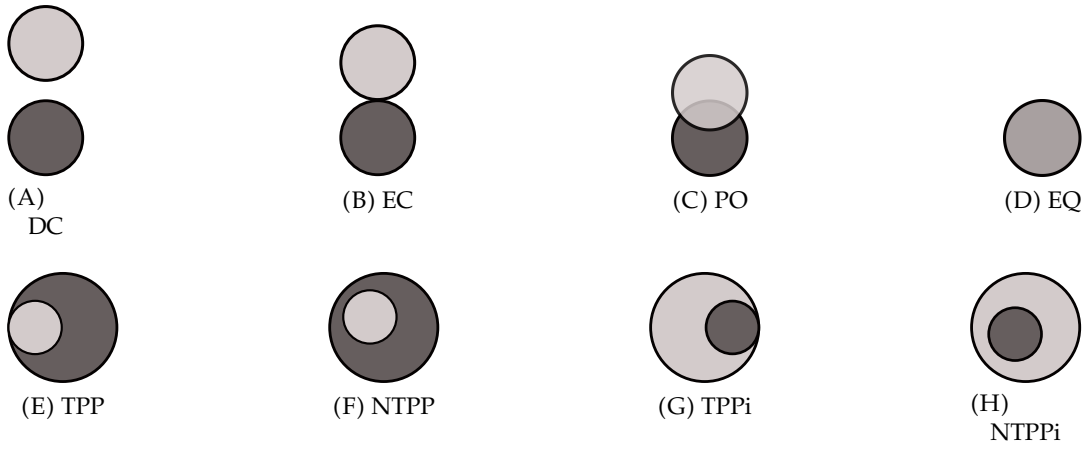


FIGURE 7.1: Pictures representing the 8 core relations of RCC8: **DC** (Disconnected), **EC** (Externally Connected), **PO** (Partially Overlaps), **EQ** (Equals), **TPP** (Tangential Partial Part), **NTPP** (Non-Tangential Partial Part), **TPPi** (Tangential Partial Part inverse) and **NTPPi** (Non-Tangential Partial Part inverse) (Randell et al., 1992).

Name	Relation	RCC definition	Fuzzy RCC definition
Disconnected	DC	$\neg C(u, v)$	$1 - C(u, v)$
Part	P	$\forall w \in \mathcal{U}, C(w, u) \Rightarrow C(w, v)$	$\inf_{w \in \mathcal{U}} \overset{t}{\rightarrow} (C(w, u), C(w, v))$
Proper Part	PP	$P(u, v) \wedge \neg P(v, u)$	$\min(P(u, v), 1 - P(v, u))$
Equals	EQ	$P(u, v) \wedge P(v, u)$	$\min(P(u, v), P(v, u))$
Overlaps	O	$\exists w \in \mathcal{U}, P(w, u) \wedge P(w, v)$	$\sup_{w \in \mathcal{U}} t(P(w, u), P(w, v))$
Discrete	D	$\neg O(u, v)$	$1 - O(u, v)$
Partially Overlaps	PO	$O(u, v) \wedge \neg P(u, v) \wedge \neg P(v, u)$	$\min(O(u, v), 1 - P(u, v), 1 - P(v, u))$
Externally Connected	EC	$C(u, v) \wedge \neg O(u, v)$	$\min(C(u, v), 1 - O(u, v))$
Non-Tangential Part	NTP	$\forall w \in \mathcal{U}, C(w, u) \Rightarrow O(w, v)$	$\inf_{w \in \mathcal{U}} \overset{t}{\rightarrow} (C(w, u), O(w, v))$
Tangential Partial Part	TPP	$PP(u, v) \wedge \neg NTP(u, v)$	$\min(PP(u, v), 1 - NTP(u, v))$
Non-Tangential PP	NTPP	$PP(u, v) \wedge NTP(u, v)$	$\min(1 - P(v, u), NTP(u, v))$

TABLE 7.1: Definition of topological relations in the original RCC (Randell et al., 1992) and the fuzzy RCC for regions u and v in a universe \mathcal{U} (Schockaert et al., 2009).

t is a left-continuous t-norm and $\overset{t}{\rightarrow}$ the residual implicator corresponding to t .

A degree of intersection and a degree of inclusion of two fuzzy sets have also been proposed. A complete review of these relations is given in (Bloch, 2005). A direct extension of the crisp definition of an intersection gives the following definition.

Definition 31: Fuzzy Degree of Intersection (Bloch, 2005)

For two fuzzy sets F and G defined on a universe \mathcal{U} with μ_F and μ_G as membership functions respectively, the *fuzzy degree of intersection* is

$$\mathcal{R}_{int}(F, G) = \sup_{u \in \mathcal{U}} t(\mu_F(u), \mu_G(u)) \quad (7.1)$$

with t a t-norm.

This represents the maximum height of the conjunction between both fuzzy sets. However, it does not account for different overlapping situations. This is why, in image understanding, the following definition, which enables to get the notion of spatial overlapping, is preferred:

Definition 32: Fuzzy Spatial Degree of Intersection (Bloch, 2005)

For two fuzzy sets F and G defined on a universe \mathcal{U} with μ_F and μ_G as membership functions respectively, the *fuzzy spatial degree of intersection* is

$$\mathcal{R}_{int}(F, G) = \frac{V_H(t(\mu_F, \mu_G))}{\min(V_H(\mu_F), V_H(\mu_G))} \quad (7.2)$$

with t a t-norm and V_H the hypervolume of a fuzzy set defined as: $V_H(\mu) = \sum_{u \in \mathcal{U}} \mu(u)$.

A fuzzy degree of inclusion can also be derived from the crisp case:

Definition 33: Fuzzy Degree of Inclusion (Bloch, 2005)

For two fuzzy sets F and G defined on a universe \mathcal{U} with μ_F and μ_G as membership functions respectively, the *fuzzy degree of inclusion* is

$$\mathcal{R}_{inc}(F, G) = \inf_{u \in \mathcal{U}} T(c(\mu_F(u)), \mu_G(u)) \quad (7.3)$$

with T a t-conorm (cf. Appendix B) and c a fuzzy complement.

7.1.2 Metric Relations

Metric relations rely on a measure: the orientation for directional relations and the distance for distance relations.

7.1.2.1 Directional Relations

Directional relations enable to express the orientation of one object relatively to a reference object. As illustrated in Figure 3.2 on page 54 and justified by (Freeman, 1975), fuzzy relations are more appropriate than crisp relations for modeling relative positions because they are intrinsically vague.

A first method consists in using angle histograms. They were introduced in (Miyajima and Ralescu, 1994b). For two objects A and R (the reference), it consists in assessing the angle between the horizontal axis and the segment linking the points p_A and p_R for all $p_A \in A$ and $p_R \in R$. An histogram is then generated from all those

angles and is normalized. In order to semantically characterize this relation, it must be compared with a known fuzzy set μ_α that represents the direction of angle α . This can be done by evaluating the compatibility (Zadeh, 1996) between the normalized histogram and μ_α . A fuzzy pattern matching approach can also be used (Cayrol et al., 1982; Dubois et al., 1988; Bloch, 2005): the necessity (pessimistic evaluation) and the possibility (optimistic evaluation) are returned computing respectively the degree of inclusion and the degree of intersection between both fuzzy sets.

A similar approach is to compute the histogram of forces (Matsakis and Wendling, 1999). It differs from the angle histogram because it takes distances into account. Objects are not only divided into points but also into longitudinal sections. It can be seen as a weighted angle histogram (Bloch, 2005). Then, templates of forces have been introduced by (Matsakis et al., 2006). They improve and generalize histograms of forces by computing to which extent each point of the space is in a given direction with respect to R . They can be represented as fuzzy landscapes (Bloch, 2005), which is an image where the value of each pixel represents to which extent it satisfies the relation. An example of fuzzy landscape is shown in Figure 7.2B. The main advantage of fuzzy-landscape-based approaches is that only one landscape has to be generated for a given relation and a given reference object. Thus, in Figure 7.2B, the relation “blue object to the right of red object” can be computed for any definition of the blue object in \mathcal{U} with a single landscape generation (corresponding to the relation “to the right of red object”).

Inspired by Newton’s law of universal gravitation, (Matsakis et al., 2009) proposed to compute the force field induced by R on its surrounding points. Then, to get to which extent one point p is in a direction δ with respect to R , the dot product between a unit vector in direction δ and the force field in p is performed. The advantages of this method are that it is less sensitive to outliers and it manages elongated and concave objects better. It can also be represented as a fuzzy landscape. (Gondra and Cabria, 2016) proposed a faster implementation of the force-field-based method. It comes from the physical observation that a force field becomes negligible from a certain distance.

Fuzzy mathematical morphology is another solution (Bloch, 1999a). Mathematical morphology relies on processing a set with another set called structuring element. The two core operators are the *erosion* and the *dilation*. Both of these operators have a fuzzy extension which can be obtained by a direct translation of crisp concepts into their fuzzy counterparts. Here, we focus on the fuzzy dilation.

Definition 34: Fuzzy Dilation (Bloch and Maitre, 1995)

For a universe \mathcal{U} , fuzzy sets F and SE defined on \mathcal{U} , the *fuzzy dilation* of F by the *structuring element* SE is noted $D_{SE}(F)$ and defined as:

$$\forall u \in \mathcal{U}, D_{SE}(F)(u) = \sup_{v \in \mathcal{U}} t(\mu_{SE}(u - v), \mu_F(v)) \quad (7.4)$$

with t a t-norm.

In the context of directional relations, the structuring element is defined as:

$$\forall u \in \mathcal{U}, \mu_{SE}^\theta(u) = \max \left[0, 1 - \frac{2}{\pi} \arccos \frac{\vec{u} \cdot \vec{v}_\theta}{\|\vec{u}\|} \right] \quad (7.5)$$

with \vec{u} the vector between the origin and u , and \vec{v}_θ the unit vector in the direction θ . Figure 7.2A shows an example of structuring element for $\theta = 0$, which represents

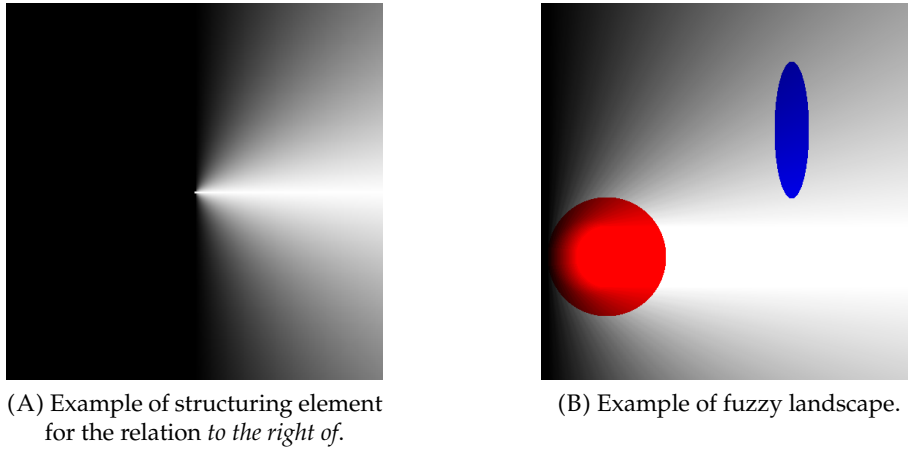


FIGURE 7.2: An example of fuzzy dilation. Figure 7.2B represents the fuzzy landscape associated to the fuzzy dialtion of the red object by the structuring element displayed in Figure 7.2A.

the relation “*to the right of*”.

The result of the dilation can be visualized as a fuzzy landscape (Bloch, 2005). An example of fuzzy landscape is displayed on figure 7.2B. It is the result of the fuzzy dilation of the red disk by the structuring element that we can see on figure 7.2A.

In figure 7.2B, in order to assess to which degree the blue ellipse is to the right of the red disk, we can use a fuzzy pattern matching approach and compute the necessity (the degree of inclusion) and the possibility (the degree of intersection) (Bloch, 1999a). A consequence of that process is that we do not need to compute another fuzzy landscape to evaluate the relation “to the right of the red disk” for any other entity in \mathcal{U} .

7.1.2.2 Distance Relations

Many distances have been proposed in the literature but most of them compare two membership functions and do not include any spatial information (Bloch, 2005). We focus on the distances that do include such information.

A first solution is to define fuzzy sets that represent a notion like “*far*” or “*near*”. Those are qualitative distance relations (Hernández et al., 1995). For example, the relation “*near*” can be expressed as follows (Schockaert et al., 2011):

$$\forall u, v \in \mathcal{U}, d_1, d_2 \in \mathbb{R}^+, N_{(d_1, d_2)}(u, v) = \begin{cases} 1 & \text{if } d_E(u, v) \leq d_1 \\ 0 & \text{if } d_E(u, v) > d_1 + d_2 \\ \frac{d_1 + d_2 - d_E(u, v)}{d_2} & \text{otherwise } (d_2 \neq 0) \end{cases} \quad (7.6)$$

with d_E the euclidean distance in \mathcal{U} . Then, the relation “*far*” can be obtained as:

$$F_{(d_1, d_2)}(u, v) = 1 - N_{(d_1, d_2)}(u, v) \quad (7.7)$$

These two relations are displayed in Figure 7.3.

Another way is to use fuzzy mathematical morphology (Bloch, 1999b). As we saw for directional relations, it is convenient for generating fuzzy landscapes. It enables to express relations such as “*at a distance less than d* ”, “*at a distance greater than d* ” and “*at a distance between d_1 and d_2* ”. The degree to which a point $u \in \mathcal{U}$ is at

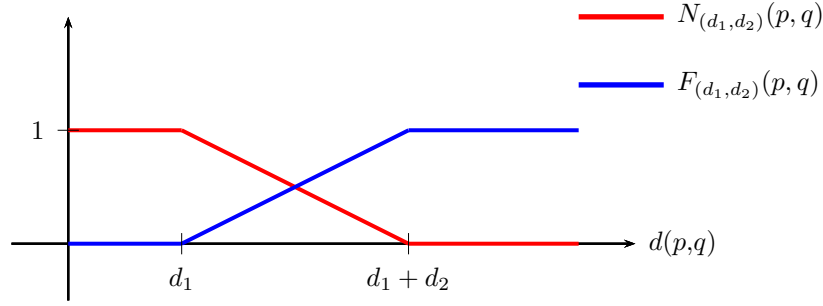


FIGURE 7.3: Example of distance relations “near” and “far”.

a distance between d_1 and d_2 from the fuzzy object F is:

$$d(u, F) = t(D_{SE_2}(F)(u), 1 - D_{SE_1}(F)(u)) \quad (7.8)$$

with t a t-norm, D_{SE_1} the dilation by the structuring element SE_1 and D_{SE_2} the dilation by the structuring element SE_2 . SE_1 and SE_2 are defined as follows (Vanegas, 2011):

$$SE_1(u) = \begin{cases} 1 - \mu_n(d_E(u, O)) & \text{if } d_E(u, O) \leq d_1, \\ 0 & \text{otherwise.} \end{cases} \quad (7.9)$$

$$SE_2(u) = \begin{cases} 1 & \text{if } d_E(u, O) \leq d_2, \\ \mu_n(d_E(u, O)) & \text{otherwise.} \end{cases} \quad (7.10)$$

with μ_n the membership function of a fuzzy set whose core (cf. Appendix B on page 171) is $[d_1, d_2]$. O is the origin of the structuring element. Thus, d is actually the conjunction of a distance smaller than d_2 (D_{SE_2}) and a distance larger than d_1 ($1 - D_{SE_1}$).

(Cinbis and Aksoy, 2007) proposed another method, which is also based on mathematical morphology. For a directional relation, we saw that the directional information is brought by the term $\frac{2}{\pi} \arccos \frac{\vec{u} \cdot \vec{v}_d}{\|\vec{u}\|}$ of the structuring element defined in Equation (7.5). For distances, instead of dealing with angular information, we can use the norm, such as:

$$\forall u \in \mathcal{U}, \mu_{SE}^\tau(u) = \max\left(0, 1 - \frac{\|\vec{u}\|}{\tau}\right) \quad (7.11)$$

with $\tau \in \mathbb{R}^{+*}$.

7.1.3 Structural Relations

Structural relations describe a pattern between two or more objects. One can see them as an extension of simpler directional relations between two objects (cf. Section 7.1.2.1).

7.1.3.1 Between

(Cinbis and Aksoy, 2007) proposed a morphological-based approach for evaluating the spatial relation “between”. Given two objects A and B , this method computes a fuzzy landscape representing to which extent each pixel is between A and B . The principle relies on computing the orientation θ_0 between A and B using a histogram

of angles. Then, the directional fuzzy landscape of A in the direction θ_0 and the directional fuzzy landscape of B in the direction $\theta_0 + \pi$ are generated. The final landscape, representing the relation “*between A and B*”, is computed as the intersection of the two previous fuzzy landscapes. An example is displayed on Figure 7.4.

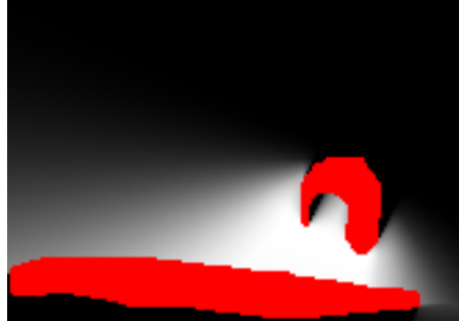


FIGURE 7.4: Example of a fuzzy landscape representing the relation “*between*” the two red objects (Cinbis and Aksoy, 2007).

7.1.3.2 Surroundedness

The “*surroundedness*” can be evaluated by looking at the angular coverage of a point or region by another region (Rosenfeld and Klette, 1985; Miyajima and Ralescu, 1994a). (Vanegas, 2011) proposed a method that generates a fuzzy landscape. For each point $u \in \mathcal{U}$, its angular coverage by a region F is computed and used to set a membership degree. It also takes into account the concavities of F so that only the points of F that see u are considered. An example is shown in Figure 7.5A.



(A) Fuzzy landscape representing the “*surroundedness*”. The purple object is the reference.



(B) Alignment of planes (in red).

FIGURE 7.5: Examples from (Vanegas, 2011).

Another solution is to use the Φ -descriptor (Matsakis et al., 2015). The Φ -descriptor of a pair of objects (A, B) is a quantitative representation of the position of A relative to B . The subregions where both objects interact in direction θ are extracted. This is done for every direction. The areas and the lengths of these subregions are computed and lead to a 28-tuple. This tuple is the descriptor. Ancillary information is then deduced from the descriptor and links to a particular relation using a template

(Francis et al., 2018). The system can return “*A is completely surrounded by B*” or “*A is somewhat surrounded by B*”. This is convenient for generating an interpretation of the relation in natural language. However, it does not compute any fuzzy landscape, so a new descriptor has to be generated for each pair of objects. It can interpret other relations, such as RCC8 relations or directional relations (Francis et al., 2018). Another method proposed by (Clément et al., 2017) can be used for assessing surroundedness. However, it mainly focuses on the assessment of the enlacement between two objects, which is detailed later.

7.1.3.3 Alignment

(Vanegas, 2011) proposed two different definitions of an alignment: global alignment and local alignment. A group of objects G is called “*globally aligned*” if:

- it contains 3 or more objects;
- all objects in G are considered as neighbours;
- all pair of objects in G are in the same direction θ .

An example of extraction of a globally aligned group of objects is displayed in Figure 7.5B.

It is called “*locally aligned*” if:

- all objects in G are considered as neighbours;
- for each object in G , its neighbours in G are aligned.

For the second point, for $A, B, C \in G$ and B and C in the neighbourhood of A , alignment is assessed by computing a degree of similarity between the angle histogram between pA and B and the angle histogram between A and C .

7.1.3.4 Enlacement

A method for assessing if two objects are *enlaced* was presented in (Clément et al., 2017). An enlacement of one object by another happens in situations similar to the ones displayed in Figure 7.6. The principle is the following: for two objects A and B , an enlacement histogram is generated for each value of the orientation. In polar coordinates, a straight line is defined by a parameter $\rho \in \mathbb{R}^+$ and a parameter $\theta \in [0; 2\pi[$. For one given orientation θ_0 , the value $E_{A,B}(\theta_0)$ in the histogram is computed as the betweenness of the longitudinal cuts A_{ρ,θ_0} and B_{ρ,θ_0} of A and B in direction θ_0 , such as (Clément et al., 2017)

$$E_{A,B}(\theta_0) = \int_0^{+\infty} \left(\int_{-\infty}^{+\infty} \mu_{A_{\rho,\theta_0}}(x) \int_x^{+\infty} \mu_{B_{\rho,\theta_0}}(y) \int_y^{+\infty} \mu_{A_{\rho,\theta_0}}(z) dx dy dz \right) d\rho \quad (7.12)$$

This enables to assess the enlacement of B by A . Then, in order to evaluate the relation “*B enlaced by A*”, a fuzzy pattern matching approach (Cayrol et al., 1982; Dubois et al., 1988; Bloch, 2005) is performed. As with angle histograms for estimating directional relations (cf. Section 7.1.2.1), the histogram we obtain is normalized and compared to a fuzzy set representing the enlacement relation. This method can also be used to assess *surroundedness*.

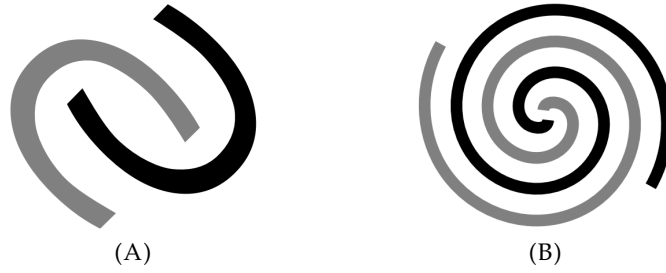


FIGURE 7.6: Examples of enlacement between two objects (Clément et al., 2017).

7.1.3.5 Parallelism

(Vanegas, 2011) introduced a way to evaluate parallelism between two objects or groups of objects. The degree of parallelism between a linear object A and an object B depends on:

- the fact that a large proportion of B should be visible to A in the direction $\theta_A + \frac{\pi}{2}$ (with θ_A the orientation of A);
- the orientation of A and the orientation of the boundary of B that is facing A and visible to A in the direction $\theta_A + \frac{\pi}{2}$ should be similar.

The definition is analogous for globally aligned groups of objects (cf. Section 7.1.3.3) in place of linear objects.

7.1.3.6 Line-region Relations

Line-region relations are relations such as “to go through” or “to intersect”. For example, for the relation “to go through”, the goal is to verify that a linear object intersects with a region and that this object does not start nor end inside the region. Thus, for a linear object L and a region R , this relation can be defined as (Vanegas, 2011):

$$\mu_{\text{go through}}(L, R) = t\left(\mathcal{R}_{\text{int}}(L, R^\circ), c(\mathcal{R}_{\text{int}}(L_s \cup L_e, R^\circ))\right) \quad (7.13)$$

with L_s and L_e the extremities of L (which is a linear object) and R° the interior of R .

7.1.3.7 Line symmetry

(Colliot, 2003) proposed a symmetry measure for fuzzy objects. For a fuzzy object A , it is defined as:

$$\sigma_A(\Pi) = \zeta(A, e_\Pi(A)) \quad (7.14)$$

with Π a plane, $e_\Pi(A)$ the reflection of A around Π and ζ a similarity measure. ζ compares two fuzzy objects and has the following properties:

1. $\zeta(A, B) = \zeta(B, A)$ (symmetric);
2. $\zeta(A, B) = 1 \Leftrightarrow A = B$;
3. $\zeta(A, B) = 0$ if and only if the support of A and the support of B are disjoint;
4. ζ is translation-invariant;

5. ζ is rotation-invariant.

For a grayscale image f , the symmetry measure around a plane Π is defined the same way:

$$\sigma_f(\Pi) = \zeta(f, e_{\Pi}(f)) \quad (7.15)$$

The only difference is that ζ now compares two images instead of two fuzzy sets. Such measures can be found in the signal processing literature and the image registration literature.

The author also proposes a method for finding the plane of symmetry in an image. It consists in 3 steps:

1. Inertia axis are computed using the covariance matrix of the image. For each inertia axis, a plane orthogonal to this axis is computed. Then, the degree of symmetry of the image around each plane is assessed;
2. The plane corresponding to the higher degree of symmetry is retained;
3. The position and the orientation of this plane is tweaked according to an optimization process. Since the derivative of the degree of symmetry is unknown, the author resorted to a derivative-free optimization method called *Nelder-Mead method* (Nelder and Mead, 1965).

It was used on 3D MRI images of the brain to compute the interhemispheric plane. Figure 7.7 contains two examples where the axis of symmetry is extracted.

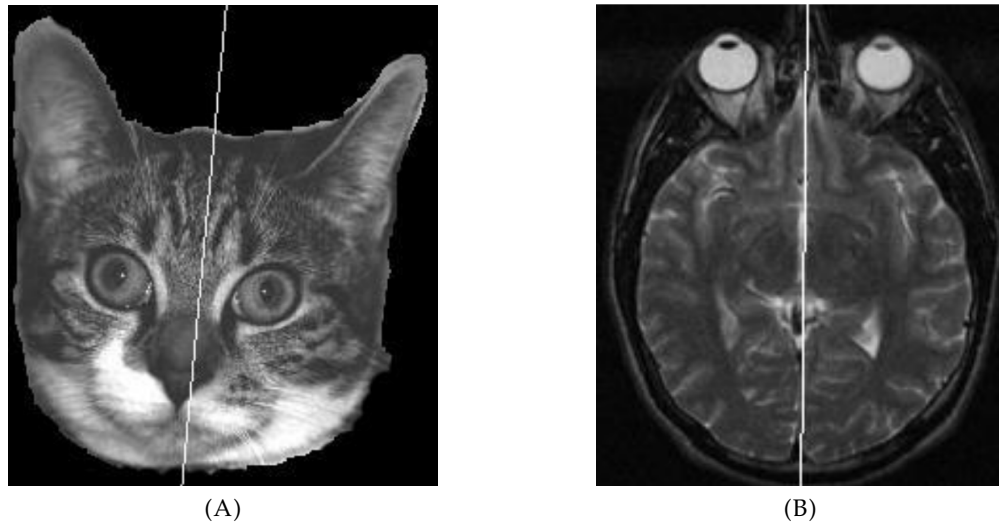


FIGURE 7.7: Axes of symmetry found by the method described in (Colliot, 2003)

7.2 Temporal Relations

Unlike fuzzy spatial relations, the literature about fuzzy temporal relations is thin. These relations enable to express vague time periods or fuzzy events. Dubois et al. (Dubois et al., 2003) presented some methods based on the possibility theory for handling imprecise and uncertain information in temporal reasoning. In particular, they proposed an extension of Allen's temporal relations (Allen, 1983), such as “before” or

“after”. (Schockaert et al., 2008a) proposed another extension of the set of qualitative relations defined by Allen so that it can deal with fuzzy intervals. The same author (Schockaert and De Cock, 2008) presents a way to deal with temporal reasoning about fuzzy time intervals as a reasoning about linear constraints. In particular, the degree to which the beginning of a fuzzy time interval A is more than d time units before the beginning of another fuzzy time interval B is defined as follows:

$$bb_d^{\ll}(A, B) = \sup_{p \in \mathbb{R}} \left(t \left(A(p), \inf_{q \in \mathbb{R}} \overset{t}{\rightarrow} (B(q), L_d^{\ll}(p, q)) \right) \right) \quad (7.16)$$

where t a t-norm, $\overset{t}{\rightarrow}$ the implication associated to t , $L_d^{\ll}(p, q) = 1$ iff $p < q - d$ and $L_d^{\ll}(p, q) = 0$ otherwise. Other similar relations comparing the beginnings and ends of A and B are defined (Schockaert and De Cock, 2008).

(Poli et al., 2016) proposed other fuzzy temporal relations, which are online as well. They rely on the notion of temporal scope, which is a temporal fuzzy set (Carriena et al., 2000) whose membership function is used to weight the different past values and to define a vague notion of past (like “recently” or “the last 5 seconds”). The authors define the following relations and properties:

- the *occurrence* of an event given a particular scope. It represents whether a certain phenomenon has occurred over a certain scope;
- the *ratio* operator, which indicates how much an expression has been true over a scope;
- the *persistence* operator, which characterizes in what extent the phenomenon is observed at each moment of the scope;
- the *precedence* for indicating whether a particular phenomenon occurred before another one;
- the *periodicity* for representing the degree to which a phenomenon is observed regularly over a certain scope.

Figure 7.8 shows examples of fuzzy temporal scopes. In (Poli et al., 2017), these relations were used for monitoring the behaviour of a wind turbine.

7.3 Spatio-temporal Relations

As fuzzy temporal relations, fuzzy spatio-temporal relations have also been little studied in the literature. (Le Yaouanc and Poli, 2012; Poli et al., 2018) present a few of them for describing the movements of geolocalized entities in the context of automatic activity recognition. In (Poli et al., 2018), the activity of a company’s vehicle fleet is monitored and described. For a mobile entity e and a region o , the following relations were used:

- the relation *IsMoving*, which evaluates the positions of the considered entity over a certain time span;
- the relation *IsComingCloseTo* between e and o . It takes into account the evolution of the position of e over a certain time span, the angle between e and o and the evolution of this angle;

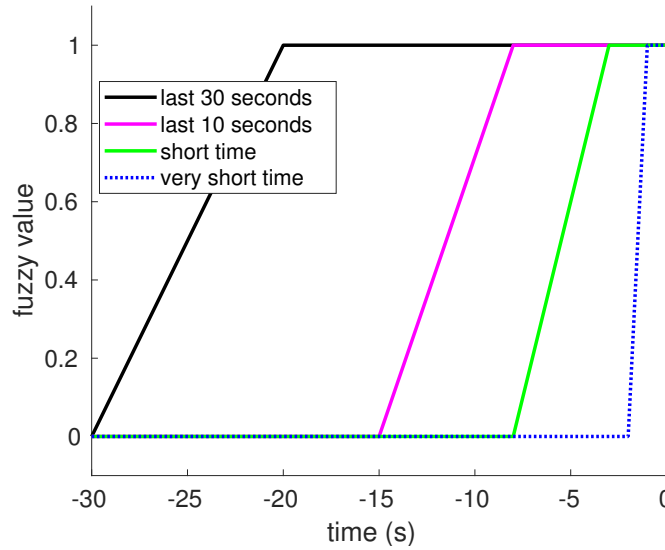


FIGURE 7.8: Examples of fuzzy temporal scopes (Poli et al., 2018).

- the relation *IsGoingAway* between e and o . It actually depends on the same variables as the relation *IsComingCloseTo*;
- the relation *IsGoingAlong* between e and o . It relies on the evolution of the location of e over a certain time span and on the proximity of e to o over this time span.

The goal of these interpretable operators is to use them for building more complex operators that are still interpretable.

7.4 Discussion

In this chapter, we presented fuzzy relations that can be used by a model built with the approach we presented in Part II. We gave an overview of fuzzy spatial relations for working on images, fuzzy temporal relations for time series and fuzzy spatio-temporal relations for problems mixing both the space and time dimensions. For a given problem, all these relations are candidates for being part of the vocabulary that is provided in our approach. This is how we make the whole system expressive. However, that does not mean that all the relations should all be included in the vocabulary. Indeed, adding relations that are bound to be irrelevant for the task we deal with will only increase the computation time of the training phase. Then, relations that were added to the vocabulary and that end up being irrelevant will be filtered by the heuristics we presented in Chapter 5.

We also saw in this chapter that the fuzzy dilation operator is versatile (directions, distances) and is often used in substeps for computing more complex relations (parallelism, alignment). Thus, it is needed for evaluating many relations. However, this operation is computationally expensive, which is why we focus in the next chapter on getting a faster implementation of it.

Chapter 8

Fast Parallel Fuzzy Morphological Operators

In Chapter 7, we introduced fuzzy relations that could be used for building a vocabulary, which is a set of relations. We saw that many different fuzzy spatial relations were proposed in the literature and that several of them rely on the fuzzy dilation operator (Bloch and Maitre, 1995). Indeed, it is a versatile operator whose properties depend on the definition of its structuring element. However, it is computationally expensive and, since it is used in the computation of several relations, it may lead to a long evaluation phase.

In Chapter 5, we presented two heuristics that make the evaluation faster. The first one, relying on the value of the minimum support, prunes the evaluation space of the relations that are bound to be infrequent. The second heuristic is based on the logical links between relations and allows to find an optimized path for evaluating relations. Both of them enable to prevent useless computations, which may take long if they resort to a fuzzy dilation. However, if we do not know whether a relation will be frequent or not, and if we have to evaluate it according to the evaluation order of relations presented in Section 5.3, then the relations involving a fuzzy dilation have to be computed. That is why we worked on another enhancement of the approach.

(Bloch, 1999a) proposed an algorithm based on a propagation technique inspired by the chamfer method (Borgefors, 1986). It returns an approximation of the fuzzy dilation. However, it does not take advantage of modern CPU architectures. (Wang et al., 2006) proposed another approximation method, based on slicing the images according to a given set of reference directions. Nevertheless, it is not a generic since the set of directions has to be set differently for different image sizes. In this chapter, we propose a new and faster implementation of the fuzzy dilation operator. It relies on two ideas:

- performing the operations in a different order to prevent useless computations,
- taking advantage of multithreading and vectorization by separating computations.

The application of these two ideas enables to get a fast execution that returns an exact result of the fuzzy dilation.

We first remind in Section 8.1 the definition of the fuzzy dilation operator and how it is used in this work. Then, in Section 8.2, we describe the three existing algorithms for computing a fuzzy dilation: the naive one, which is a direct translation of the definition, the algorithm proposed by (Bloch, 1999a) and the algorithm proposed by (Wang et al., 2006). Section 8.3 is dedicated to the new algorithm we propose in this thesis. In Section 8.4, we present the experiments that we carried out and the results we got. Finally, we discuss these results in the last section.

8.1 Fuzzy Dilation

Like the dilation operator in mathematical morphology, the fuzzy dilation operator is the result of set-theoretic operations between an input image (representing the reference object for the dilation) and a structuring element. It was formally defined in Definition 34 on page 108. For the sake of clarity, this definition is reminded here. For a universe \mathcal{U} , fuzzy sets F and SE defined on \mathcal{U} , the *fuzzy dilation* of F by the *structuring element* SE is noted $D_{SE}(F)$ and defined as:

$$\forall u \in \mathcal{U}, D_{SE}(F)(u) = \sup_{v \in \mathcal{U}} t(\mu_{SE}(u - v), \mu_F(v)) \quad (8.1)$$

with t a t-norm. An example of fuzzy dilation is displayed in Figure 8.1. It enables to generate a fuzzy landscape that can be then used for evaluating how close to F any object in \mathcal{U} is.

Several t-norms are defined in the literature. In this paper, we use the *Zadeh* t-norm, which is the *minimum*. Besides, since the fuzzy dilation is applied on images, \mathcal{U} is a finite set so the *supremum* is equivalent to the *maximum*. Thus, the expression of the fuzzy dilation we actually implemented is the following

$$\forall u \in \mathcal{U}, D_{SE}(F)(u) = \max_{v \in \mathcal{U}} \min(\mu_{SE}(u - v), \mu_F(v)) \quad (8.2)$$

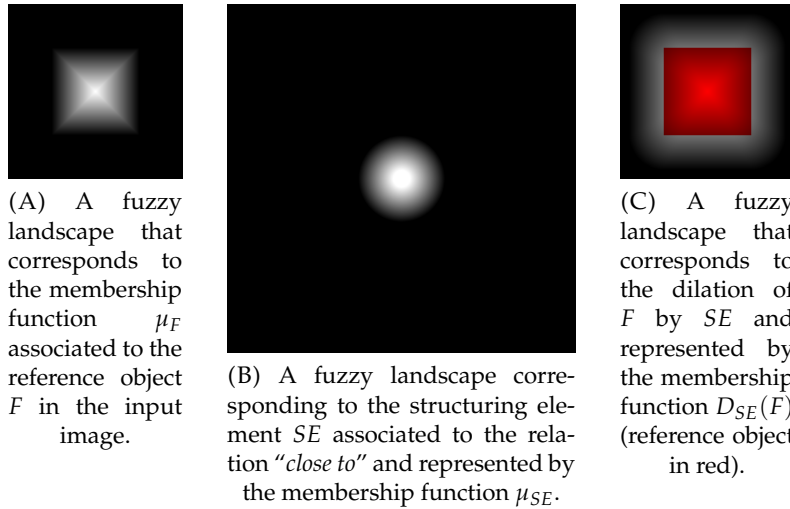


FIGURE 8.1: Fuzzy dilation $D_{SE}(F)$ of the reference object F by the structuring element SE . The spatial relation represented here is "close to". The intensity of each pixel of $D_v(\mu)$ represents in which extent it verifies the relation. The image of the structuring element is four times as big as the two others because it is needed for performing the dilation.

8.2 Related Algorithms

In this section, we provide and describe the three existing algorithms for performing a fuzzy dilation. The first algorithm we present is the direct translation of the mathematical definition. The second one, proposed by (Bloch, 1999a), relies on a

propagation method that returns an approximation. The third one approximates the fuzzy dilation by considering only the contributions in a given set of directions.

8.2.1 Naive Forward Algorithm

The *naive* algorithm is the direct application of Equation (8.2). Three data structures (cf. Figure 8.1) are involved:

- F , which is the input image containing the reference object and whose size is $N \times M$,
- SE , which is the structuring element of the dilation and whose size is $2N \times 2M$,
- $D_{SE}(F)$, which is the fuzzy dilation of F by SE and whose size is $N \times M$.

This algorithm is shown in Algorithm 8. It computes the value of each pixel in $D_{SE}(F)$ in one forward pass, from the top-left pixel to the down-right one. However, for each iteration, the machine has to loop over the whole input image F while applying the structuring element SE on it, regardless of the actual number of pixels belonging to the reference object ($\mu_F(u) > 0$). Thus, $(NM)^2$ max/min operations have to be performed leading to a high computation time.

Algorithm 8: Naive algorithm for performing the fuzzy dilation of the reference object F by the structuring element SE .

input :

- the input image F corresponding to the reference object
- the structuring element SE

output: the fuzzy dilation $D_{SE}(F)$ of F by SE

```

1 forall  $u \in \mathcal{U}$  do
2    $D_{SE}(F)(u) \leftarrow 0$ 
3   forall  $v \in \mathcal{U}$  do
4      $val \leftarrow \mu_F(v)$ 
5      $se \leftarrow \mu_{SE}(u - v)$ 
6      $D_{SE}(F)(u) \leftarrow \max(\min(val, se), D_{SE}(F)(u))$ 
7   end
8 end
9 return  $D_{SE}(F)$ 
```



FIGURE 8.2: For each iteration of the fuzzy dilation, $D_{SE}(F)(u)$ with $u \in \mathcal{U}$, the structuring element SE is looped over the input image F . As shown on these two figures, SE needs to be bigger than F to cover it completely.

The structuring element SE must be four times as big as the input image because of the term $\mu_{SE}(u - v)$ in Equation (8.2). It means that μ_{SE} is centered in v and thus it needs to be four times as big as F when v is a border pixel of the input image. This is represented in Figure 8.2.

Furthermore, while $D_{SE}(F)$ and F are scanned forward, SE is scanned backward leading to an inefficient CPU cache usage and an inefficient loop vectorization.

Therefore, this algorithm is computationally very expensive. In practice, it is never used since a faster algorithm was proposed by (Bloch, 1999a).

8.2.2 Bloch's Propagation Algorithm

(Bloch, 1999a) proposed an algorithm based on a propagation technique inspired by the chamfer method (Borgefors, 1986). It returns an approximation of the fuzzy dilation.

This algorithm is displayed in Algorithm 9. First, the functions ϕ and \angle are defined. Then, on line 3, an array P is instantiated to store intermediary results. In the first loop (from line 4 to line 12), $\tilde{D}_{SE}(F)$ and P are initialized. Then, the propagation is performed in two steps: one forward pass and one backward pass on the image. The advantage of this method is that it relies on a neighbourhood of the current point (lines 14 and 21) and not on the full image as in the exhaustive method. Each pass updates the values in $\tilde{D}_{SE}(F)$. Once the passes have been performed, the result is returned.

Algorithm 9 (on the following page) enables to perform a fuzzy directional dilation but one should just change ϕ to compute another relation.

Using the same notations as in the previous subsection, the time complexity of this method is $O((1 + 2N_N)NM)$ with N_N the size of the neighbourhood.

As previously said, the result generated by this method is an approximation of the result we would get with the naive method. Figure 8.3 compares the results we get for both methods with two different reference objects. While in most cases this difference is low, it can be perceptible to the human eye in a few others cases, as in Figure 8.3F. We carried out an experiment to quantify the impact of this difference on the final evaluation of a relation. As we said in Chapter 7, for fuzzy landscapes, a relation between two objects is evaluated using a fuzzy pattern matching approach. That is what we did on a dataset of organs that we present more exhaustively in Chapter 9. For this experiment, we had 35 images containing each 9 different organs. We computed four directional relations between all possible pairs of organs: "to the left of", "to the right of", "above" and "below". That makes a total of 10080 relations to evaluate between organs. We analyzed the absolute value of the difference between the evaluation with the exact fuzzy landscape and the evaluation with the approximated one. We found that the mean absolute error is equal to $3.98 \cdot 10^{-3}$, which is approximately the degree of precision of our images. Indeed, since each fuzzy landscape is represented as a 8-bit grayscale image, this degree equals $\frac{1}{255} \approx 3.92 \cdot 10^{-3}$. Also, we deal with vague concepts so an imprecision of this magnitude should not affect the performance of the approach. Therefore, our conclusion is that the approximated method has little impact on the relations we evaluate.

However, this algorithm cannot be parallelized since it relies on a propagation method. Indeed, the propagation is gradually spread over the image to update each pixel. That is why it is not suited to a multithreaded approach. In the following section, we present an enhanced version of the naive algorithm that is fully compatible with parallel computing.

Algorithm 9: Bloch's propagation algorithm for returning an approximation of the fuzzy dilation of the reference object F by the structuring element SE (Bloch, 1999a).

input :

- the input image F corresponding to the reference object
- the structuring element SE

output: an approximation $\tilde{D}_{SE}(F)$ of the fuzzy dilation of F by SE

- 1 Let $\phi: [0; 2\pi] \rightarrow [0; 1]$ be a function such that
 $\phi(x) = \max(0, 1 - \frac{2x}{\pi}), \forall x \in [0; 2\pi]$
- 2 Let $\angle: \mathcal{U}^2 \rightarrow [0; 2\pi]$ be a function such that $\angle(u, v)$ is the angle between the vector from the origin to u and the vector from the origin to v
- 3 Let P be a new array defined on \mathcal{U}
- 4 **forall** $u \in \mathcal{U}$ **do**
- 5 $\tilde{D}_{SE}(F)(u) \leftarrow 0$
- 6 **if** $\mu_F(u) > 0$ **then**
- 7 $P(u) \leftarrow u$
- 8 **end**
- 9 **else**
- 10 $P(u) \leftarrow -1$
- 11 **end**
- 12 **end**
- 13 **forall** $u \in \mathcal{U}$ **do** // forward pass
- 14 Let $\mathcal{N}(u)$ be the neighbourhood of u
- 15 $\mathcal{N}^*(u) \leftarrow \mathcal{N}(u) \setminus \{v \in \mathcal{N}(u) \mid P(v) = -1\}$
- 16 $v_{\max} \leftarrow \operatorname{argmax}_{v \in \mathcal{N}^*(u)} \left(\min \left(\mu_F(P(v)), \phi(\angle(u, P(v))) \right) \right)$
- 17 $\tilde{D}_{SE}(F)(u) \leftarrow \min \left(\mu_F(P(v_{\max})), \phi(\angle(u, P(v_{\max}))) \right)$
- 18 $P(u) = P(v_{\max})$
- 19 **end**
- 20 **forall** $u \in \mathcal{U}$ in the reverse order of the forward pass **do** // backward pass
- 21 Let $\mathcal{N}(u)$ be the neighbourhood of u
- 22 $\mathcal{N}^*(u) \leftarrow \mathcal{N}(u) \setminus \{v \in \mathcal{N}(u) \mid P(v) = -1\}$
- 23 $v_{\max} \leftarrow \operatorname{argmax}_{v \in \mathcal{N}^*(u)} \left(\min \left(\mu_F(P(v)), \phi(\angle(u, P(v))) \right) \right)$
- 24 $\tilde{D}_{SE}(F)(u) \leftarrow \min \left(\mu_F(P(v_{\max})), \phi(\angle(u, P(v_{\max}))) \right)$
- 25 $P(u) = P(v_{\max})$
- 26 **end**
- 27 **return** $\tilde{D}_{SE}(F)$

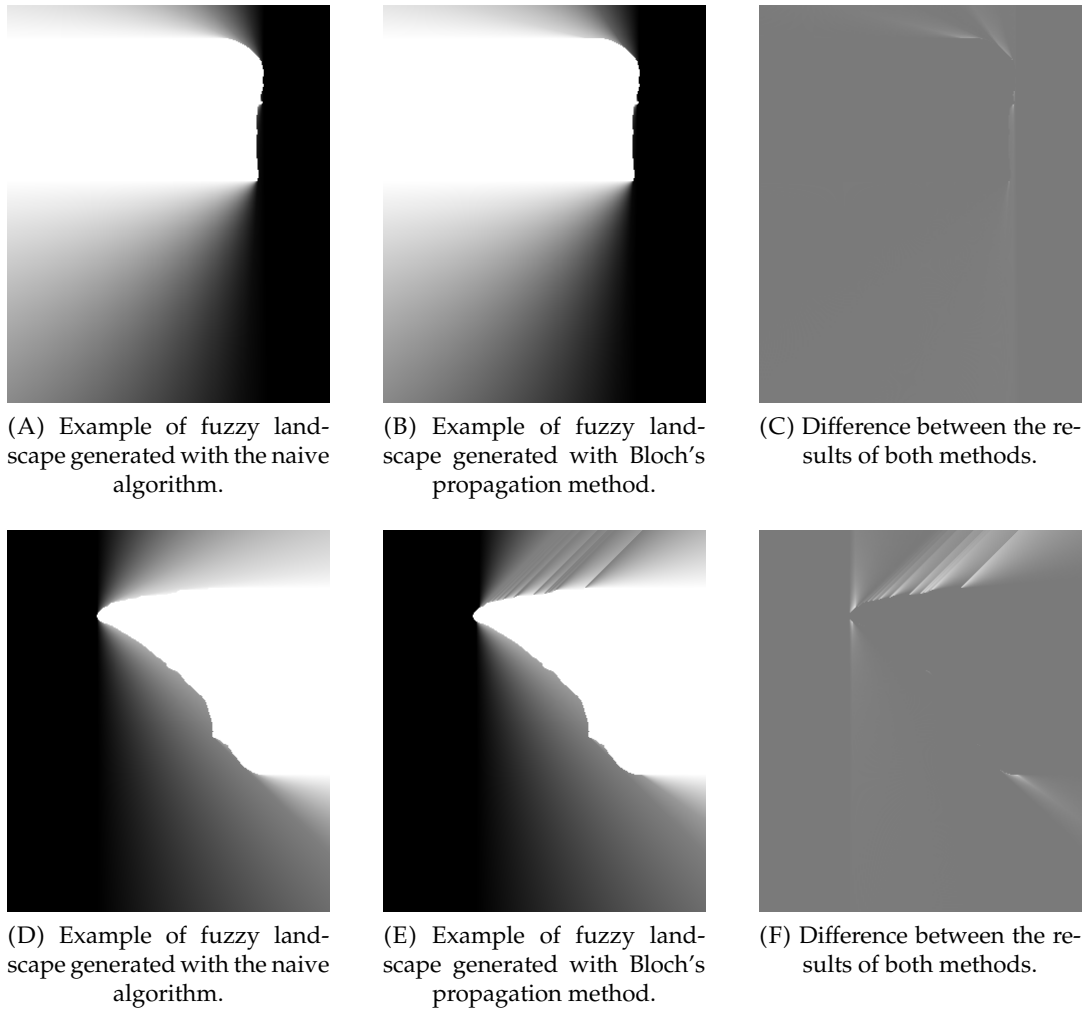


FIGURE 8.3: Comparisons of fuzzy landscapes generated with both the naive algorithm and Bloch's method.

8.2.3 Wang's Algorithm

Wang proposed an algorithm based on F-templates (Wang et al., 2006; Matsakis et al., 2006) that provides an approximation of the fuzzy dilation. The main idea is to only take into account the contribution of pixels in a set of K reference directions. Thus, for each pixel p , only the pixels on straight lines going through p in one of the reference directions contribute to the final result.

While this method is faster than Bloch's algorithm for small images (100×100) and when K is low (90), Gondra and Cabria showed that K must actually be proportional to \sqrt{MN} to keep a good approximation (Gondra and Cabria, 2016). So K must be greater for bigger images to have a correct approximation. For example, $K = 90$ is not convenient for a 400×400 image (Gondra and Cabria, 2016). This tradeoff between accuracy and speed quickly turns to the advantage of *Bloch*. Besides, maintaining the same accuracy requires to set a new value of K each time the size of the input image changes, which regularly happens in the medical image dataset that we worked on and we present in Section 9.2.1 on page 140. Thus, this algorithm is not as generic as the two others and we discarded it from our experiments.

8.3 Vectorized Multithreaded Reverse Algorithm

This section is dedicated to the new algorithm we propose for computing a fuzzy dilation. We first present a new algorithm for computing the fuzzy dilation, the *reverse* algorithm, which is based on a high level algorithmic transformation that reduce the amount of computations. Then, we explain how we implemented this algorithm using vectorization and multithreading.

8.3.1 Reverse Algorithm

The idea of the *reverse* algorithm, displayed in Algorithm 10, is to reorder the operations to avoid unnecessary processing: a pixel with a zero value in the input image (lines 2 and 3) does not contribute to $D_{SE}(F)$ (cf. Equation (8.2)) since $\min(\mu_{SE}(v - u), \mu_F(v)) = 0$ if $\mu_F(v) = 0$. That is the difference with the *naïve* algorithm where those pixels cannot be separated from the contributing ones. This can be represented as:

$$\begin{aligned} &\forall v \in \mathcal{U} \text{ such as } \mu_F(v) > 0, \\ &D_{SE}(F)(u) = \max \left(\min(\mu_{SE}(u - v), \mu_F(v)), D_{SE}(F)(u) \right) \forall u \in \mathcal{U} \end{aligned} \quad (8.3)$$

Algorithm 10: Reverse algorithm for performing the fuzzy dilation of the reference object F by the structuring element SE .

```

input :
  • the input image  $F$  corresponding to the reference object
  • the structuring element  $SE$ 
output: the fuzzy dilation  $D_{SE}(F)$  of  $F$  by  $SE$ 

1 forall  $v \in \mathcal{U}$  do
2    $val \leftarrow \mu_F(v)$ 
3   if  $val > 0$  then
4     forall  $u \in \mathcal{U}$  do
5        $se \leftarrow \mu_{SE}(u - v)$ 
6        $D_{SE}(F)(u) \leftarrow \max(\min(val, se), D_{SE}(F)(u))$ 
7     end
8   end
9 end
10 return  $D_{SE}(F)$ 

```

Instead of looping over the pixels of $D_{SE}(F)$ in the main loop, we loop over the pixels of the input image F . This enables to detect and drop all the computations related to these non-contributing pixels. Thus, each contributing pixel of the input image is read only once. Then, its contribution to the dilation is evaluated over $D_{SE}(F)$ with the structuring element centered around the input pixel position (from line 4 to line 7). This difference is shown on Figure 8.4, where the executions of both the naïve algorithm and the reverse algorithm are illustrated.

Due to the associative properties of the min and max operators, the final result is exactly the same as with the *naïve* algorithm. The number of pixels in the reference object directly affects the number of max/min operations, so the processing time for computing the fuzzy dilation should mainly depend on the size of the reference

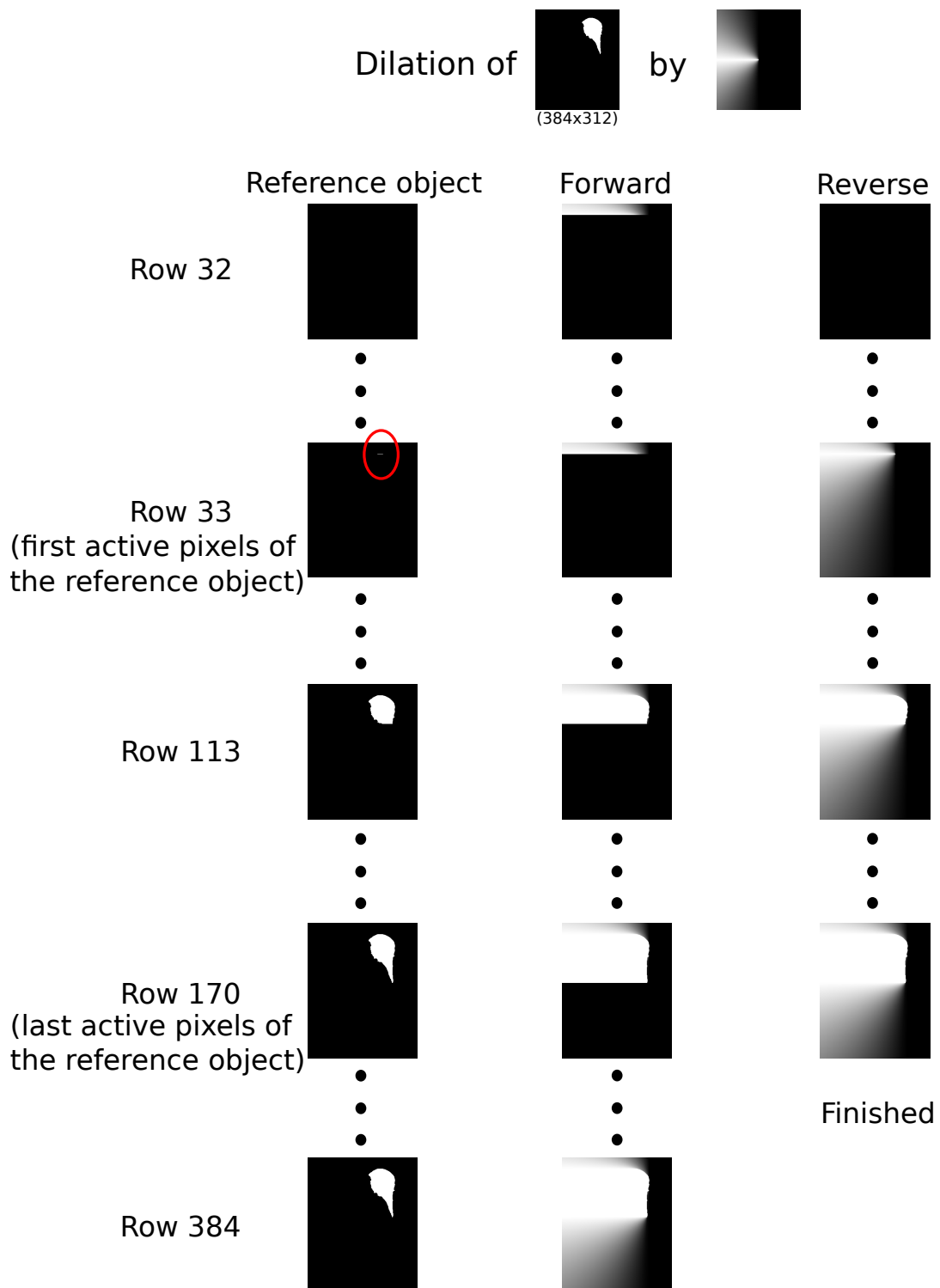


FIGURE 8.4: Comparison of the naive and reverse algorithms for computing a fuzzy dilation. The input image has 384 rows. While the naive approach computes the dilation pixel by pixel, the reverse one only computes the contributions of the non-zero pixels in the reference object. On row 33, the first non-zero pixels of the reference object are barely perceptible on the figure and have been surrounded by a red ellipse. Once the reference object has been completely looped over (row 170), the reverse algorithm finishes and returns the same result as the naive approach after row 384.

object. For an object of size p pixels belonging to a $N \times M$ input image, only $p \times NM$ max/min operations are executed, providing an acceleration factor dependent on the object size.

Furthermore, the structuring element SE and the result of the dilation $D_{SE}(F)$ are both scanned forward, which induces a better cache usage and a direct SIMD (cf. Appendix G on page 185) alignment.

8.3.2 Multithreading and Vectorization

High-level code transformations, such that the reverse algorithm, conjugated with the full usage of the multiple cores and the vector instructions (SIMD) offered by modern CPU architectures have proven their efficiency (Lacassagne et al., 2014). Thus, we propose algorithms that take advantage of these.

8.3.2.1 PR: parallel reverse algorithm

While the *reverse* algorithm processes only the active pixels, it still computes their contribution pixel by pixel on one core. As seen in Algorithm 10 on line 6, with the reverse modification, the contribution of one pixel of value $val = \mu - F(v)$ to the output $D_{SE}(F)$ consists in applying only separable and aligned operations based on $D_{SE}(F)$ and the structuring element centered on the current active pixel. Then, for each pixel where $\mu_F(v) > 0$ (active pixels), the *parallel reverse* (PR) algorithm uses the OpenMP parallelization framework (Dagum and Menon, 1998) to dispatch the computations of $D_{SE}(F)$ over each core using a strip based spatial decomposition as shown in Figure 8.5. The internal loop (from line 4 to line 7 in Algorithm 10) is processed in parallel using the `#pragma omp for` directive. By construction, the *reverse* algorithm prevents data races as each strip of $D_{SE}(F)$ is processed by a different thread and SE is only accessed in read mode. To avoid recreating the threads for each new active pixel, which would harm the overall performance, parallel regions are created before the actual parallelization using the `#pragma omp parallel` directive over the external loop (line 1 in Algorithm 10).

8.3.2.2 SIMD Optimizations: PR_{128} , PR_{256} and PR_{512}

In addition to the PR algorithm, we propose 3 SIMD implementations using explicit SIMD instructions¹:

- PR_{128} is the same as PR except that it uses explicit AVX SIMD instructions. This enables to compute one operation on 16 8-bit unsigned integers.
- PR_{256} is specifically designed for the *de facto* standard version AVX2 (256-bits wide instructions). This enables to compute one operation on 32 8-bit unsigned integers.
- PR_{512} resorts to the recent AVX512 extension (512-bits wide instructions). 64 8-bit unsigned integers can be treated at the same time.

The way computations are dispatched combining OpenMP and SIMD instructions is displayed in Figure 8.5.

PR , PR_{128} , PR_{256} and PR_{512} can be easily modified to use either 8-bits, 16-bits, 32-bits integers, float or double.

¹The reader can find more details about SIMD in Appendix G on page 185

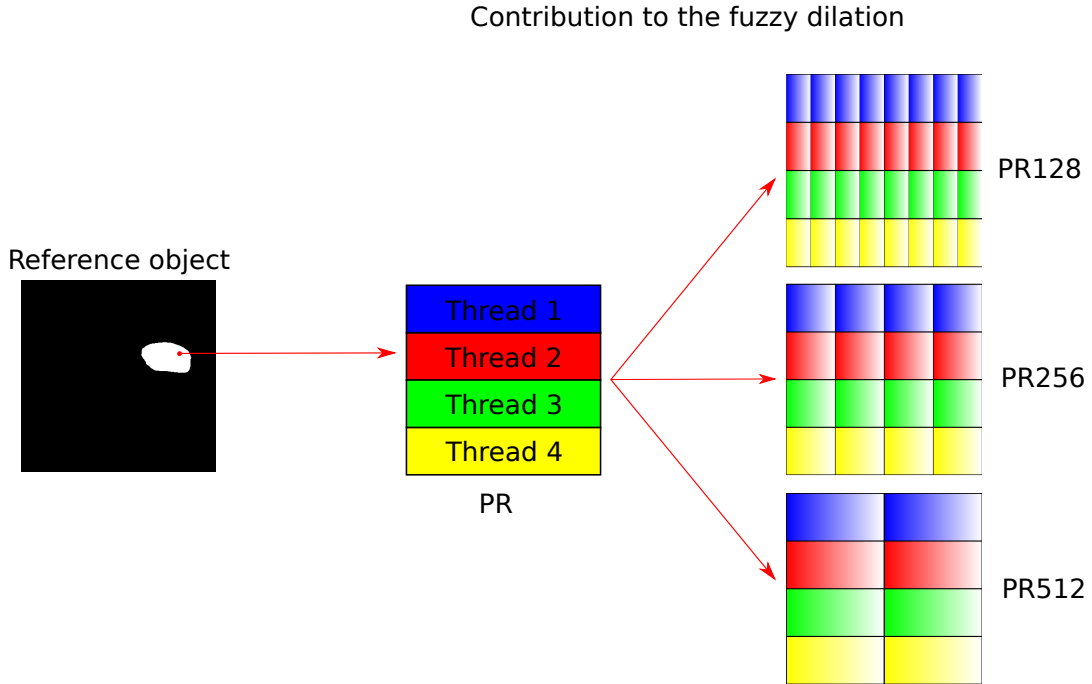


FIGURE 8.5: Contribution to the fuzzy dilation of one active pixel (in red) from the reference object (on the left). For the sake of this illustration, only 4 threads and 128 columns are represented. PR is the multi-threaded version of the *reverse* algorithm where each thread is responsible for a strip of rows of the fuzzy dilation $D_v(\mu)$. With PR_{128} , PR_{256} and PR_{512} , for each thread, columns of $D_v(\mu)$ are distributed using AVX, AVX2 and AVX512 respectively.

8.4 Benchmark and Results

8.4.1 Dataset and Benchmark Configurations

The benchmark we propose consists in computing the fuzzy dilation corresponding to the relation “to the right of” the reference object. Other dilation-based operations were not considered since they only rely on another structuring element.

Seven algorithms were evaluated: the *naïve* algorithm, the *reverse* algorithm, its parallel version PR and its three SIMD implementations PR , PR_{128} , PR_{256} and PR_{512} , and *Bloch's* algorithm. We evaluated the algorithms efficiency, using an extensive benchmark on two different datasets:

- A structured artificial dataset of 282 images. This dataset provides crisp and fuzzy images, round, rectangle and ellipsoidal shapes, a regular distribution of squares, different sizes of images (256×256 , 512×512 and 1024×1024) and different sizes of reference objects (from 1 pixel to 65536 pixels). A few samples from this dataset are shown in Figure 8.6.
- A dataset of medical images coming from (Jimenez-del-Toro et al., 2016) (more details about this dataset can be found in Section 9.2.1 on page 140). We selected ten 312×384 images (from 1234 active pixels to 6138, so that is 1.0% to 5.1% of the image) and eight 407×1515 images (from 1096 active pixels to 9112, so that is 0.2% to 1.4%). Those images correspond to non-fuzzy segmented organs like the reference objects in Figure 8.4 and Figure 8.5.

The artificial dataset enables to evaluate algorithms on specific criteria while the dataset of medical images is more representative of real world applications.

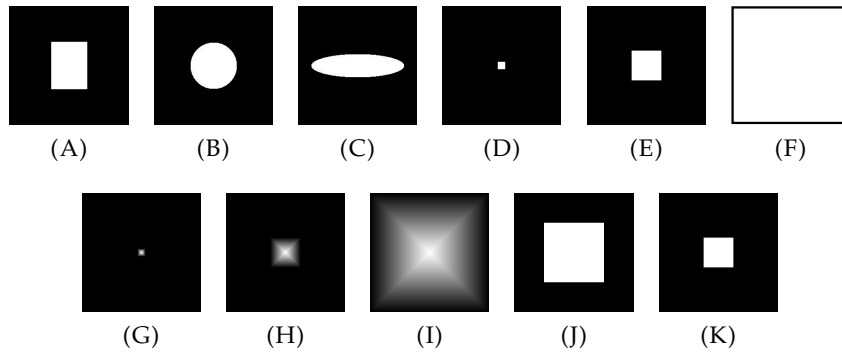


FIGURE 8.6: Artificial dataset samples: a) Rectangle, b) Disk, c) Ellipse, d) 256×256 crisp square with 256 active pixels, e) 256×256 crisp square with 4096 active pixels, f) 256×256 crisp square with 65536 active pixels, g) 256×256 fuzzy square with 256 active pixels, h) 256×256 fuzzy square with 4096 active pixels, i) 256×256 fuzzy square with 65536 active pixels, j) 512×512 crisp square with 65536 active pixels, k) 1024×1024 crisp square with 65536 active pixels

All the computations were performed on an Intel Xeon CPU 6148 (20 cores - fixed 2.4GHz frequency) using executables generated by the Intel ICC compiler 2019.3.

8.4.2 Results

Using the artificial dataset, the benchmark confirms that the fuzziness, the shape and the position of the reference object in the input image have no effect on the execution time ($< 1\%$ of execution time variation). As expected, the image size, the reference object size and the algorithm are the three relevant parameters. Table 8.1 and Table 8.2 present the execution time of the seven algorithms for three image sizes and two reference object sizes: 4096 (referred as *small objects*) and 65536 (referred as *large objects*) respectively. Table 8.3 presents the acceleration ratio of each algorithm using the *reverse* algorithm as the reference (the *naive* algorithm is too slow to be the baseline, which would skew the comparisons).

Sequential algorithms In this paragraph, we focus on naive, reverse and Bloch's algorithms. We observe that the *naive* algorithm is always the slowest one. This algorithm is not suitable for time-realistic object relationship evaluation since it would take about three days to generate all the fuzzy landscapes for a dataset of 35 images with 9 entities per image and 4 different directions. As expected, due to its construction, the execution time is mainly related to the image size and not to the reference object size. The *reverse* algorithm is faster in every configuration. When the reference object size increases, the execution time of *reverse* increases but still remains faster than *naive* as it provides an additional acceleration factor (xs) compared to the expected *active vs total* pixel ratio. This additional factor is underscored in Table 8.2 for a 256×256 image where $xs = 2.1$. Indeed, in this configuration, all pixels are active and both *forward* and *reverse* compute the exact same number of pixels. The fact that *reverse* uses both the output $D_{SE}(F)$ and the structuring element SE in a cache-friendly way is beneficial.

In every configuration, Bloch's algorithm is faster than *reverse* with a speedup of $\times 11$ for small objects and $\times 170$ for large objects.

Algorithm	256×256	512×512	1024×1024
Naive	11.9×10^3	191×10^3	3075×10^3
Reverse	354.1	1.4×10^3	5.6×10^3
<i>PR</i>	55.2	82.4	297.7
<i>PR</i> ₁₂₈	9.8	12.6	27.1
<i>PR</i> ₂₅₆	9.0	10.7	22.2
<i>PR</i> ₅₁₂	8.4	9.8	19.5
Bloch	33.0	130.9	523.1

TABLE 8.1: Execution time in ms for one fuzzy landscape computation with variable image sizes and a 4096-pixel centered fuzzy square. Bold font indicates the best result for each case.

Algorithm	256×256	512×512	1024×1024
Naive	11.8×10^3	189×10^3	3040×10^3
Reverse	5.6×10^3	22.5×10^3	89.7×10^3
<i>PR</i>	454.1	1307.3	4720.7
<i>PR</i> ₁₂₈	144.8	183.9	407.9
<i>PR</i> ₂₅₆	137.8	162.4	319.5
<i>PR</i> ₅₁₂	124.5	147.4	277.5
Bloch	35.4	134.6	530.9

TABLE 8.2: Execution time in ms for one fuzzy landscape computation with variable image sizes and a 65536-pixel centered fuzzy square. Bold font indicates the best result for each case.

Algorithm	256×256	512×512	1024×1024
4096-pixels centered object			
Naive	3.0×10^{-2}	7.4×10^{-3}	1.8×10^{-3}
Reverse	1	1	1
<i>PR</i>	6.4×10^0	1.7×10^1	1.9×10^1
<i>PR</i> ₁₂₈	3.6×10^1	1.1×10^2	2.1×10^2
<i>PR</i> ₂₅₆	3.9×10^1	1.3×10^2	2.5×10^2
<i>PR</i> ₅₁₂	4.2×10^1	1.4×10^2	2.9×10^2
Bloch	1.1×10^1	1.1×10^1	1.1×10^1
65536-pixels centered object			
Forward	4.8×10^{-1}	1.2×10^{-1}	3.0×10^{-2}
Reverse	1	1	1
<i>PR</i>	1.2×10^1	1.7×10^1	1.9×10^1
<i>PR</i> ₁₂₈	3.9×10^1	1.2×10^2	2.2×10^2
<i>PR</i> ₂₅₆	4.1×10^1	1.4×10^2	2.8×10^2
<i>PR</i> ₅₁₂	4.5×10^1	1.5×10^2	3.2×10^2
Bloch	1.6×10^2	1.7×10^2	1.7×10^2

TABLE 8.3: Speedups with the *reverse* algorithm as reference for variable image and reference object sizes. Bold font indicates the best result for each case.

Unlike *reverse*, Bloch’s algorithm computation time does not depend on the reference object size. However, similarly to *reverse*, it linearly increases along the image size. This is due to their complexity: $O((1 + 2N_N)NM)$ for Bloch’s algorithm and $O(pNM)$ for *reverse*.

SIMD multi-thread algorithms For small objects (cf. Table 8.1), PR_{512} is the fastest in every configuration. For large objects (cf. Table 8.2), Bloch’s algorithm is the fastest for smaller images, while PR_{512} is the fastest for 1024×1024 images. This is explained by the fact that the execution time of PR and its derivatives does not follow the same image size progression as *reverse* and Bloch’s since they are significantly more efficient for larger images (cf. Table 8.3).

Using all 20 cores of our benchmark processor, PR_{512} is faster than *reverse* by a factor $f_R \in [42; 320]$ and by $f_{PR} \in [6; 18]$ compared to PR . However, using the maximum number of core is not always the optimal solution in our context. As shown in Table 8.4, for a sufficient amount of data (large object and large image), PR_{512} with all cores is the best solution. However, for smaller images and objects, the best option is to use less cores (8 for a 4096-pixels object in a 256×256 image and 16 for a 4096-pixels object in a 512×512 image). In all cases, despite the combined pressure of the SIMD extensions and multiple threads on memory bandwidth, the SIMD natural order is respected as $t_{PR_{512}} < t_{PR_{256}} < t_{PR_{128}} < t_{PR}$. Thus, PR_{512} is the best implementation of the PR algorithm.

Active cores	2	4	8	16	20
65536-pixels object in a 1024×1024 image					
PR_{128}	3933.3	1587.2	833.3	461.2	407.9
PR_{256}	3044.4	1140.4	608.4	354.4	319.5
PR_{512}	2746.9	978.8	509.3	306.0	277.5
4096-pixels object in a 512×512 image					
PR_{128}	50.1	27.0	16.4	11.8	12.6
PR_{256}	36.6	20.3	12.9	9.8	10.7
PR_{512}	30.5	17.2	11.8	9.1	9.8
4096-pixels object in a 256×256 image					
PR_{128}	14.8	9.9	7.3	7.3	9.8
PR_{256}	10.7	7.6	6.4	6.7	9.0
PR_{512}	9.6	7.2	5.9	6.7	8.4

TABLE 8.4: Execution time in ms for one fuzzy landscape computation with variable number of active cores and reference object sizes.

Natural images Real-world images confirms PR_{512} as the fastest dilation operator algorithm. In Figure 8.7, the low object size dependency of Bloch’s is highlighted by close results for each image category. On these images, even the PR algorithm is competitive with Bloch’s as the number of active pixels with respect to the image size is low (cf. Section 8.4.1). Both the accelerating effect of AVX extensions and their limitations when data are not sufficient are well illustrated. Indeed as shown in Table 8.5, for the 407×1515 images, the algorithms keeps accelerating along AVX,

AVX2, and AVX512 modifications. This effect is lower or non-existent for 312×384 images.

	312 × 384			407 × 1515		
Alg.	min	mean	max	min	mean	max
<i>PR</i>	×1.0	×2.2	×4.8	×0.8	×3.3	×6.7
<i>PR</i> ₁₂₈	×4.1	×9.4	×21.5	×7.7	×26.6	×51.0
<i>PR</i> ₂₅₆	×4.4	×9.8	×20.8	×9.1	×30.6	×57.3
<i>PR</i> ₅₁₂	×5.0	× 10.6	×21.5	×10.1	× 33.5	×61.6

TABLE 8.5: Acceleration ratio with *Bloch*’s algorithm as reference with variable image and reference object sizes.

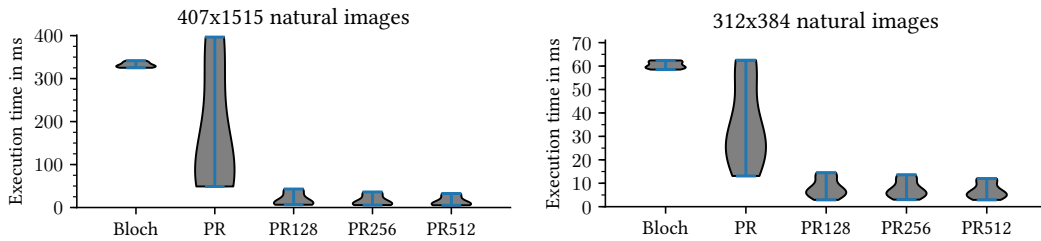


FIGURE 8.7: Execution time distributions for the natural image dataset.

8.5 Discussion

Since the fuzzy dilation is computationally expensive and it is involved in the computation of several fuzzy spatial relations that could be part of a vocabulary, we focused in this chapter on a new faster implementation of this operation. Starting from the mathematical definition of the fuzzy dilation, we studied the naive implementation of this operator in order to improve it. We proposed a new algorithm that only takes into account the pixels that contribute to the final result. Also, it enables to loop forward over the structuring element so that it minimizes cache misses. We measured that our solution is faster than the state-of-the-art method on real world images and remains competitive in each situation of our artificial dataset. It is best suited for small reference objects while it can be slower than Bloch’s algorithm for bigger reference objects. Overall, for the sizes of the objects we will treat in the next chapter, the solution we proposed is the fastest in the literature.

Although we only tackled the fuzzy dilation, other fuzzy morphological operators such as the erosion, the opening and the closing can be optimized similarly.

8.6 Acknowledgements

We achieved the work presented in this chapter in collaboration with Laurent Cabaret from the research laboratory in Mathematics and Computer Science (MICS) in CentraleSupélec.

This work was performed using HPC resources from the “Mésocentre” computing center of CentraleSupélec and École Normale Supérieure Paris-Saclay supported by CNRS and Région Île-de-France.

Chapter 9

Experiments on Images

In this chapter, we present the experiments that we carried out to evaluate the approach described in Part II on images: given a vocabulary of relations, the most frequent of them are extracted in order to generate rules/constraints to perform the target task and produce explanations. Relying on the work we presented in the two previous chapters, a vocabulary of relations can be set for the two experiments we present in this chapter.

We evaluated three different aspects of the models built with our approach:

- Its raw performance in terms of *accuracy*, for assessing how well the model recognizes examples or entities.
- The *explanations it provides*, for evaluating how suited to the task the explanations generated by the model are.
- How efficiently the heuristics we proposed in chapter 5 on page 71 *prune the evaluation space*. We assess the proportion of relations that were not evaluated because of these heuristics.

We carried out two experiments on image datasets. The first one, that we present in Section 9.1, consisted in building a model that should classify images from a toy dataset. Four classes of images were generated so that any image can be classified relying on the spatial arrangement of the entities that it contains. Then, in Section 9.2, the goal is to annotate organs in medical images. In this example, the system learns relational representations for each type of organ in order to generate constraints and define a FCSP.

9.1 Toy Dataset for Image Classification

This example is the first example we worked on in order to test the approach we presented in the previous chapters.

In this section, we first present the toy dataset we built and the relations and properties that belong to the catalogue. Then, we describe more accurately the different steps of the workflow and we present the results we got.

9.1.1 Dataset

In this toy dataset of images, each example contains three fuzzy shapes: a square, a disk and an ellipse. These shapes correspond to the entities that will be handled in this experiment. Images from this dataset are divided into four classes. The difference between the classes is the spatial distribution of the fuzzy shapes. The shapes in each image of the same class are similarly spatially distributed. Examples from

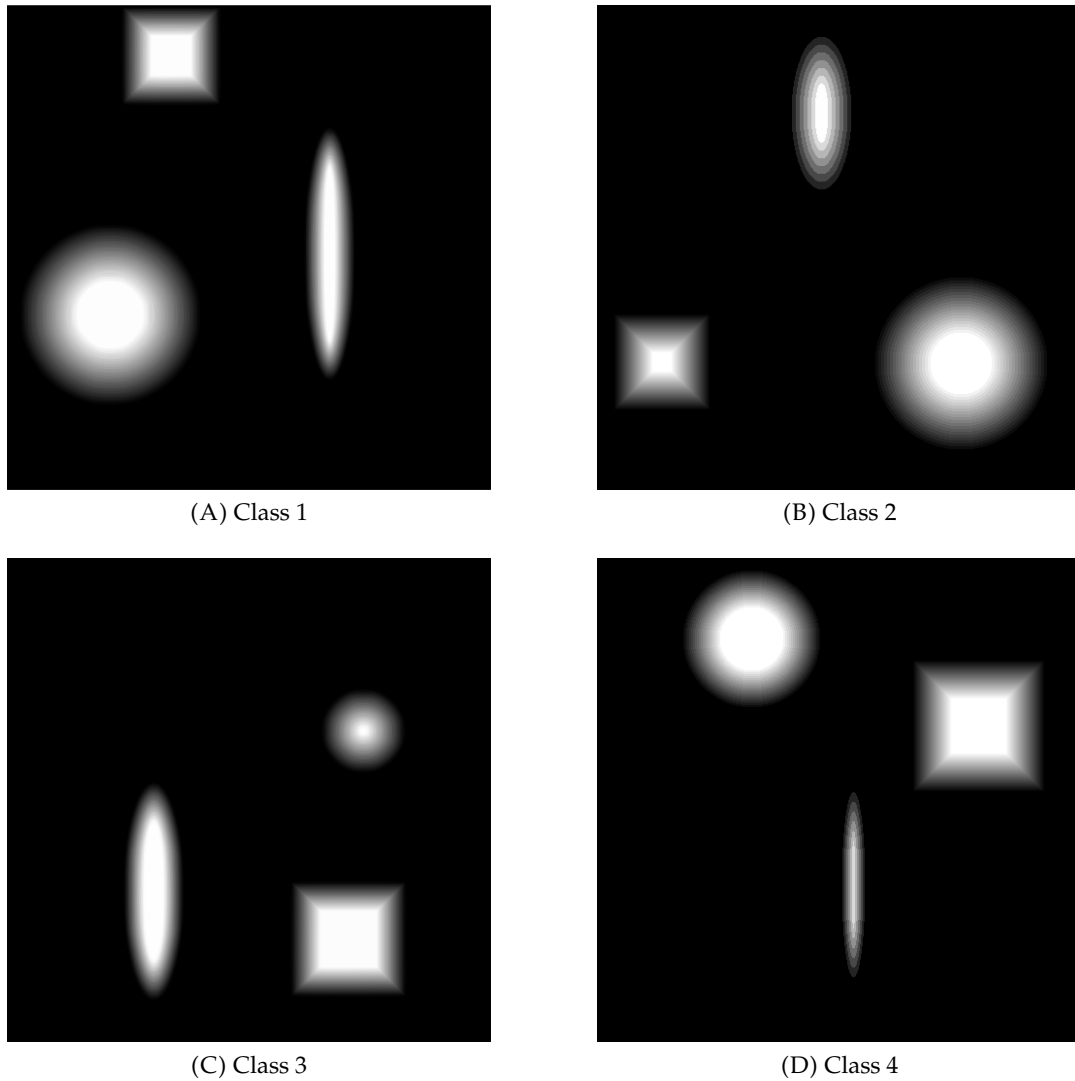


FIGURE 9.1: Examples from each class of the dataset used in the example of explained classification in Section 9.1

each class are shown in Figure 9.1. The dimensions and the fuzziness of each shape in each image vary independently of the class. Also, four borderline examples have been added to each class in order to test the reliability of the system. Each class contains 44 images and so the whole dataset is composed of 176 examples. The resolution of these images is 500×500 .

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

According to the way this dataset has been built, the expected explanation for justifying why an example belongs to a particular class should mention the relative position of fuzzy entities to each other. The relative distance between each entity varying independently of the class, it should not be part of the explanation.

9.1.2 Vocabulary of Relations

Based on the way the dataset has been built, we limited the dyadic relations that are used in this experiment to directional relations. In particular, those are fuzzy directional dilations (Bloch, 1999a) that were described in chapter 7 on page 105 and chapter 8 on page 117. We use four of them: *to the left of* ($\theta = \pi$), *above* ($\theta = \frac{\pi}{2}$), *to the right of* ($\theta = 0$) and *below* ($\theta = -\frac{\pi}{2}$). Figure 9.2D shows how the relation *ellipse to the right of disk* is evaluated.

The unary relations we selected are shape-related. One of them assesses how close or far to a disk the shape of an entity is, another one assesses how close or far to a square the shape is and a third one assesses how close or far to an ellipse the shape is. We call them *is disk*, *is square* and *is ellipse*.

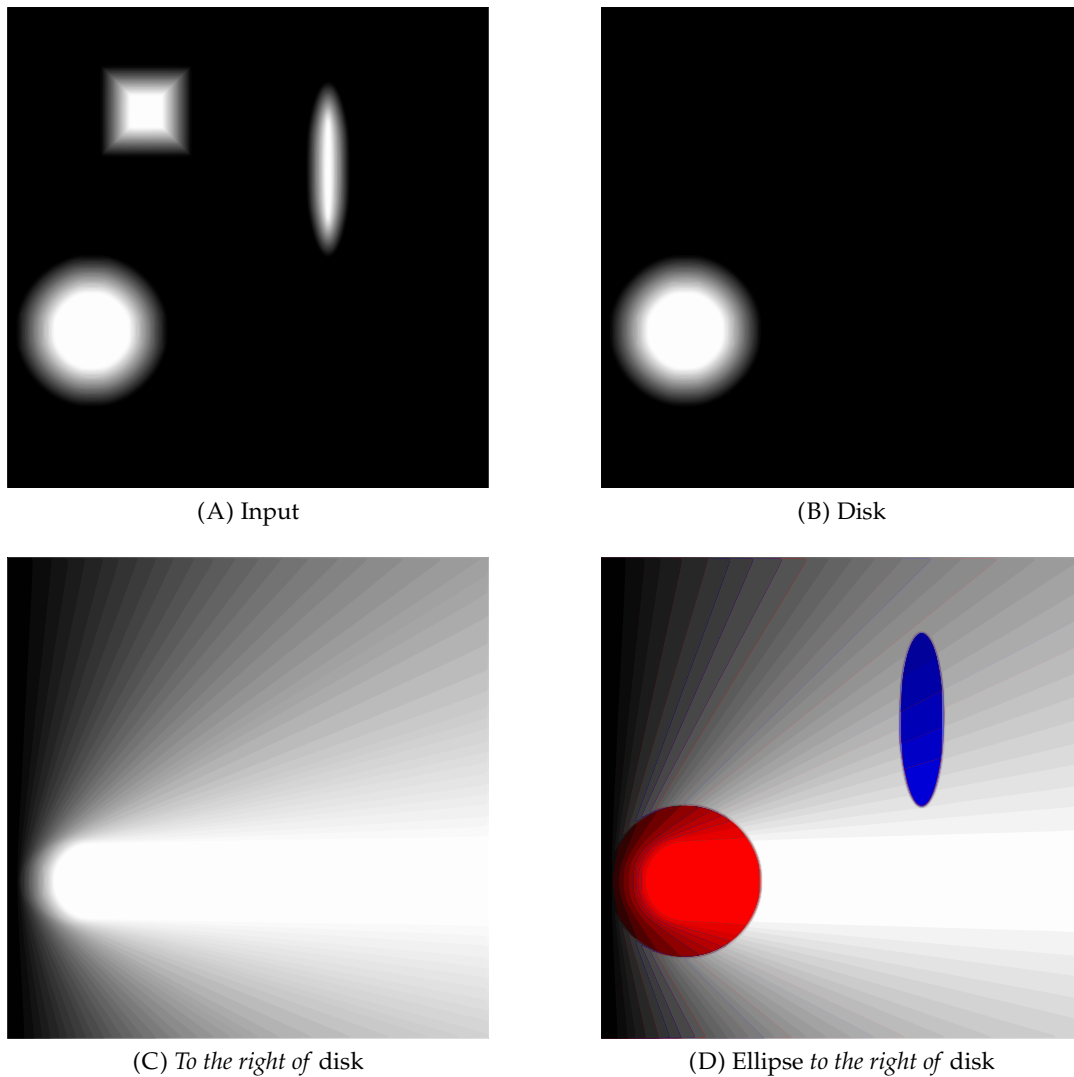


FIGURE 9.2: Example of how a specific relation is computed in an instance. Here, the goal is to compute the relation *ellipse to the right of disk*. Given an instance (Figure 9.2A), the disk, which is the reference object in the relation to evaluate, is extracted (Figure 9.2B). The fuzzy landscape *to the right of disk* can then be computed (Figure 9.2C). Finally, as explained in Chapter 7, the relation can be evaluated using a fuzzy pattern matching approach.

(Chanussot et al., 2005) presented an extension to fuzzy objects of the shape signature based on the distance of boundary points to the centroid of the object. It is based on averaging the signatures over the alpha cuts (cf. Appendix B on page 171) of the fuzzy object. We use this signature to build our three unary relations. Figure 9.3 displays an example of such a signature for a fuzzy ellipse.

Let χ be the signature of an entity. The relation *is disk* is defined as:

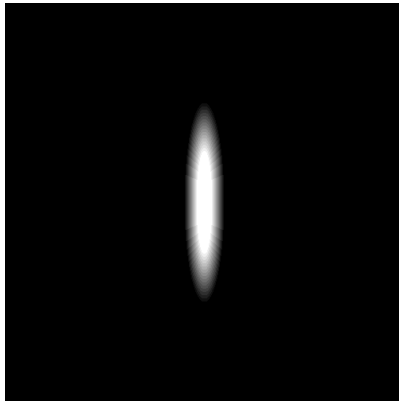
$$isDisk(\chi) = \begin{cases} 1 - \Delta & \text{if } \Delta \leq 1 \\ 0 & \text{otherwise} \end{cases} \quad (9.1)$$

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

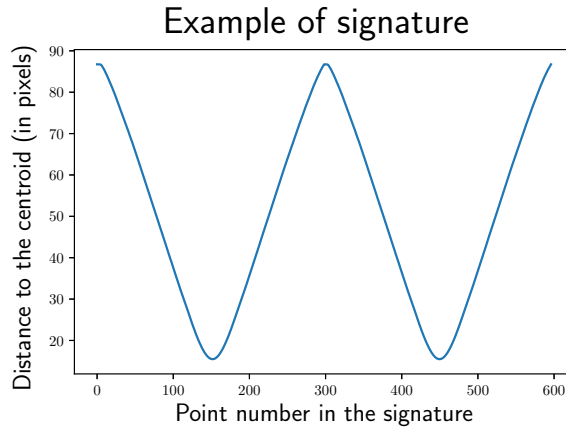
$$isSquare(\chi) = |\text{corr}(\chi, \chi_{\text{ref}})| \quad (9.2)$$

isEllipse is defined the same way. These reference shapes are a perfect square and a perfect ellipse for *is square* and *is ellipse* respectively.

Overall, we have 4 different spatial dyadic relations and 3 different unary relations. As there are 3 different entities in an instance, the total number of relations to evaluate per instance is equal to 33 (according to property 1 on page 51). Based on how the dataset was built, these relations should be sufficient for classifying examples correctly.



(A) Fuzzy ellipse



(B) Signature of the fuzzy ellipse

FIGURE 9.3: Example of signature of a fuzzy ellipse generated using (Chanussot et al., 2005). For each alpha cut of the ellipse, its edge is extracted and the distance of each point of this edge to the centroid of the ellipse is computed. Thus, we get a shape signature for each alpha cut. At the end, the final signature is obtained by averaging all the signatures we got at the previous step. Here, we can see that there are two maxima and two minima of same values. Also, these extrema are uniformly spread over the edge. This signature is typical of an ellipse.

9.1.3 Ordering of Relations

In order to get an ordering of relations that enables to prevent some useless computations, the logical links between relations from the vocabulary are represented in a graph. This graph is displayed in Figure 9.4. The edges between directional relations hold because of the specific shapes of the entities present in the dataset.

Relying on the heuristic presented in Section 5.3, we applied Kahn's algorithm (Kahn, 1962) (cf. Appendix E on page 181) to get the following order:

$$\text{above} \rightarrow \text{below} \rightarrow \text{to the left of} \rightarrow \text{to the right of} \rightarrow \text{is disk} \rightarrow \text{is square} \rightarrow \text{is ellipse} \quad (9.3)$$

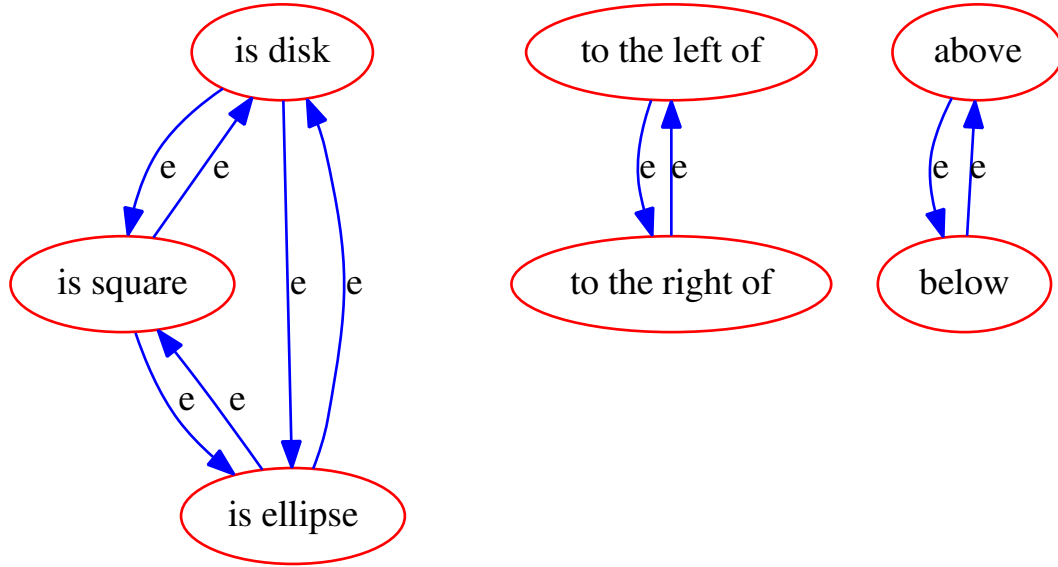


FIGURE 9.4: Graph representing the logical links between the relations in the vocabulary. There are three subgraphs with only one type of edge: e , which represents $\mathcal{R}_1 \Rightarrow \mathcal{R}_2$.

9.1.4 Workflow

This first experiment allows us to go into further detail regarding the whole workflow of the approach we propose in this thesis. For the sake of clarity, we refer to each entity in the inputs by their shape. However, the model does not know which entity corresponds to which shape until shape-related relations are assessed. The workflow is as follows:

1. During this step, all the relations from the vocabulary are assessed on the entities of each instance in the training set. Relations are evaluated according to the order given in Section 9.1.3. Also, the heuristic presented in Section 5.2 is applied to discard infrequent relations online.
2. In the second step, we would like to extract the most frequent subsets of relations among the ones that have been assessed in step 1. In order to perform this task, we split the training set into 4 subsets (one for each class). We then apply the fuzzy Close algorithm for each subset. Thus, we obtain one or several sets of relevant relations for each class.
3. For each class, we have one or more subsets of relevant relations. We can build fuzzy relational rules based on those. We already know that the consequent of

those rules is the class. Then, as all our relations are linguistically interpretable, we can predict the class of an instance and provide an explanation for this decision.

9.1.5 Results

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

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

The results we got are displayed in Figure 9.5. We manage to reach an accuracy of 100% when the value of the minimum support ranges from 0.5 to 0.65. This corresponds to the range of values where rules that were pruned from relations common to other classes reach a maximum length of 6. This observation seems logical: pruned rules contain relations that are class-specific, so longer pruned rules should have a better discriminative power.

We verify that the length of unpruned rules increases when the minimum support decreases. However, pruned rules do not follow the same behaviour. When the value of the minimum support is too low, many relations are considered frequent

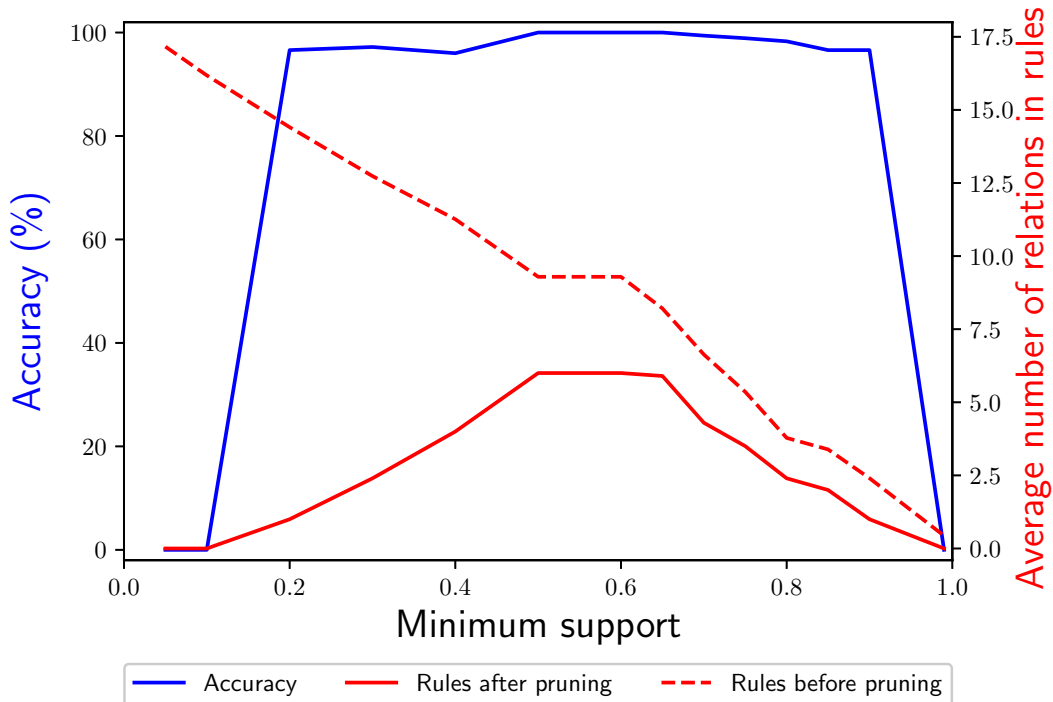


FIGURE 9.5: Plot displaying the performance of the model with respect to the value of the minimum support. We reach an accuracy of 100% between 0.5 and 0.65. The red dashed line represents the average number of relations in rules before they were pruned from relations common to other classes. The red plain line is its counterpart for rules that were pruned from common relations.

and thus rules from all classes share many common relations. Therefore, the pruning remove most of these relations, which are not relevant. That is actually a case of overfitting, since too many irrelevant relations are learnt.

On the other hand, when the value of the minimum support is too high, only few frequent relations are extracted. The risk is that those relations are common to all classes and thus cannot be used for classification. This is a case of underfitting. When all the relations in a rule are shared by some rules from other classes, the pruning turns this rule into an empty rule. This is why, in Figure 9.5, the average length of pruned rules is equal to 0 when the minimum support is set to 1.

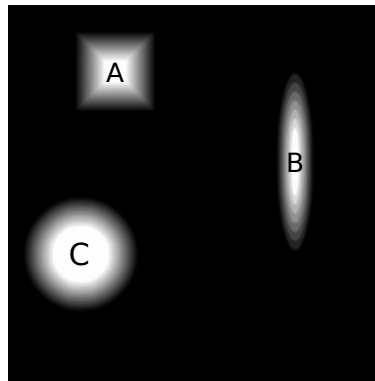
Setting a different vocabulary could enable to get relations whose evaluation is closer to 1. For example, we could achieve this using directional relations that are closer to the orientation between entities. However, those relations are also less easy to translate into natural language than the four core directions.

Figures 9.6 to 9.9 show an example of explained classification for class 1, 2, 3 and 4 respectively. We can see that the most obvious relations are used for explaining the output. Moreover, one can notice in the example of Figure 9.6 that the relations *object B is below object A* and *object A is above object B* express the same situation. That seems obvious that if one of these two relations is used, the other one will be too. However, that may not always be the case due to the way these relations are computed. Indeed, when we use fuzzy mathematical morphology, the result depends on the shape of the reference object. So two reciprocal relations may not have the same evaluation. That is why there are slight differences that can impact the relations involved in the antecedent of the rule.

Figure 9.9 presents an interesting borderline case. This instance belongs to class 4 and has been classified as a member of this class. However, this decision has been made with less confidence than non-borderline instances, which are easier to classify. Indeed, the relation *object A to the left of object B* is less satisfied in this instance than in other instances from class 4. Also, the relations that were learnt for class 3 entails a non-negligible firing of the corresponding rules (cf. Figure 9.8). In the end, we get a confidence of 0.32 for class 4, which we qualify as *average*, against 0.17 for class 3.

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

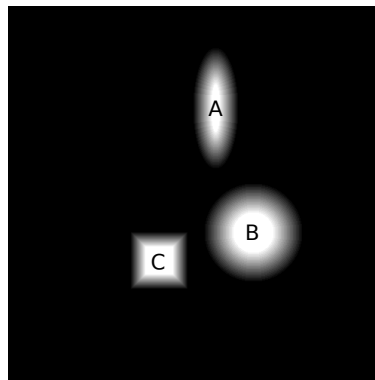
However, our second heuristic, which consists in using the logical links between relations, is not suited for this experiment. Indeed, there are no relations depending on others nor symmetrical ones. In the graph displayed in Figure 9.4, we represented the implications between a few relations. As we explained in Chapter 5, implications are useful when a relation is fully satisfied or not at all. Here, there are only implications that requires relations to be fully satisfied. Overall, only one relation has been evaluated to 1: this is the relation *object C below object A* in Figure 9.9. This enabled us to deduce that the relation *object C above object A* was equal to 0 before assessing it.



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

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

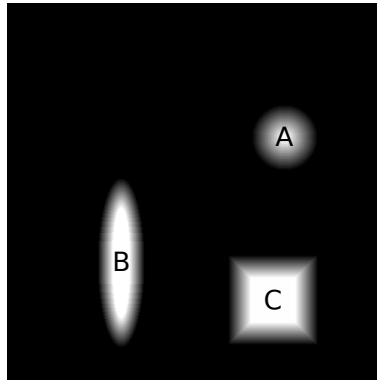
FIGURE 9.6: Example of explanation for an instance from class 1. Relations in bold were pruned because they are shared by rules from other classes.



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

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

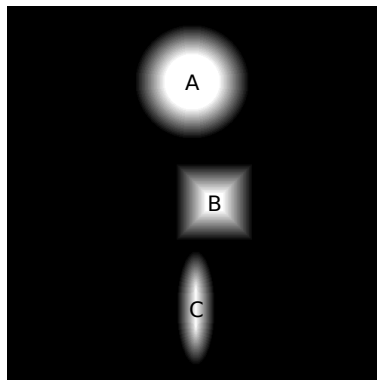
FIGURE 9.7: Example of explanation for an instance from class 2. This is one of the borderline example we generated. Relations in bold were pruned because they are shared by rules from other classes.



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

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

FIGURE 9.8: Example of explanation for an instance from class 3. Relations in bold were pruned because they are shared by rules from other classes.



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

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

FIGURE 9.9: Example of explanation for an instance from class 4. This is one of the borderline example we generated. Relations in bold were pruned because they are shared by rules from other classes.

9.2 Organ Annotation in Medical Images

In this experiment, we aim at performing explained automatic image annotation. It is performed on real images, so that will give us more clues about the performance of our approach in a more realistic situation.

In this section, we first present the dataset, which contains crisp entities. Then, we describe the relations that are in the vocabulary to solve this problem and we present the workflow of the approach in the case of annotation. The following subsection presents the results we obtained. Finally, we tackle the evaluation of the explanations we generated in this experiment.

9.2.1 Dataset

The original dataset we use here comes from the VISual Concept Extraction challenge in RADioLogY (VISCERAL) project (Langs et al., 2013). This project proposes a cloud-based infrastructure for the evaluation of medical image analysis techniques in Computed Tomography (CT) and Magnetic Resonance Imaging (MRI). It contains one dataset called *Anatomy3* (Jimenez-del-Toro et al., 2016). This dataset is used in one segmentation benchmark. It is composed of 391 CT and MRI images:

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

Figure 9.10 displays one example for each category of scan. Those are all 3D images that are actually the superposition of 2D slices. As we work on 2D images, we consider only slices in the following.

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

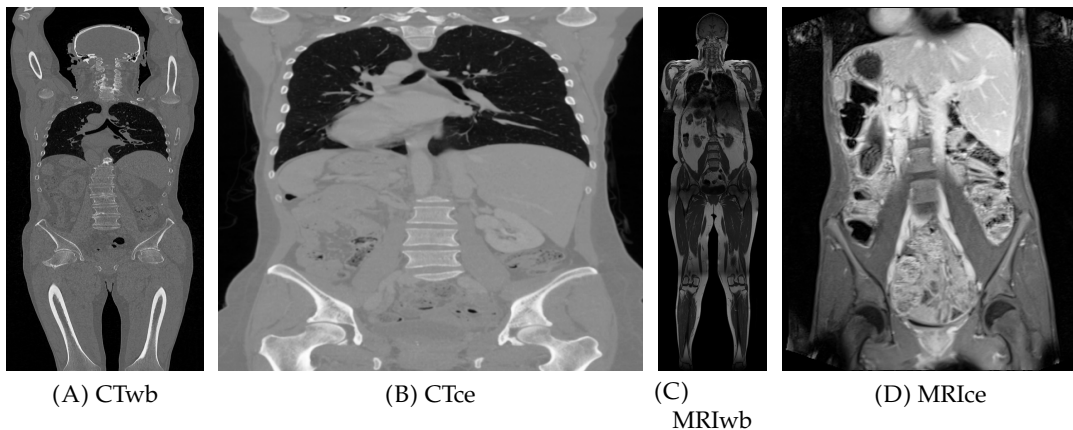


FIGURE 9.10: Examples of the four types of scans in the dataset.

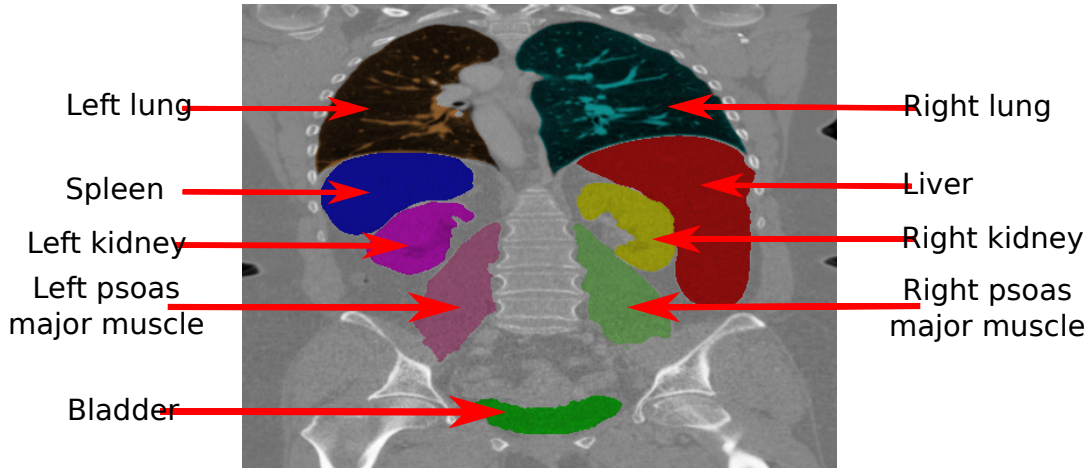


FIGURE 9.11: In this experiment, we consider the 9 colored organs in this figure.

From these images, we created our own dataset for the purpose of this experiment. Since most images from the original dataset contain few segmented organs, we selected the instances containing the 9 following organs: the *liver*, the *spleen*, the *urinary bladder*, the *left and right kidneys*, the *left and right lungs* and the *left and right psoas major muscles*. That makes 35 images and 315 segments. These nine organs are represented in Figure 9.11. The goal will be to label them in each instance. This new dataset is small, which enables to assess how our model can perform by learning from few examples.

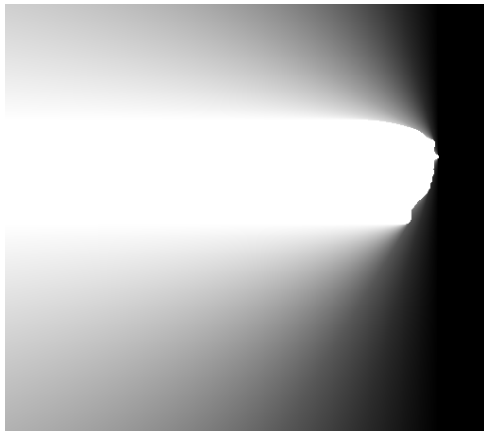
Images from Anatomy3 are NIfTI files that contains 3D scans. In our case, since we work on 2D images, we selected one slice for each of the 35 images in our dataset. Depending on the modality and the patient, slices are different, so we picked them manually so that they contain the organs we are interested in.

9.2.2 Vocabulary of Relations

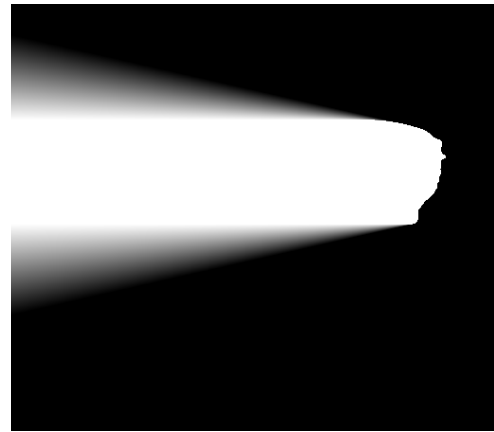
In this example, we used 9 relations. Four of them are the same directional relations (Bloch, 1999a) as we used in the previous experiment: *to the left of*, *above*, *to the right of* and *below*.

Four other relations are also directional. They express the same directions (left, right, above, below) but their fuzzy landscapes cover a smaller area of the image to express the following relations: *completely to the left of*, *completely to the right of*, *completely above* and *completely below*. They can be obtained by tweaking the structuring element (cf. Equation (7.5) on page 108). We replaced the factor $\frac{2}{\pi}$ by $\frac{14}{\pi}$, which gives the result displayed in Figure 9.12. Thus, these relations are less often satisfied and enable to express more accurate situations.

The last relation is the symmetry measure that is presented in (Colliot, 2003). The goal of this operator is to assess if two organs are symmetrical. It should be relevant since our dataset contains three pairs of organs, as shown in Figure 9.13A: lungs, kidneys and psoas major muscles. This operator is applied on an image that includes two organs. As we want to evaluate how symmetrical two organs are, we tweaked the original operator so that the line of symmetry cannot intersect with one organ. That case can happen when both organs are different and one of them is symmetrical. Figure 9.13B shows such an example.

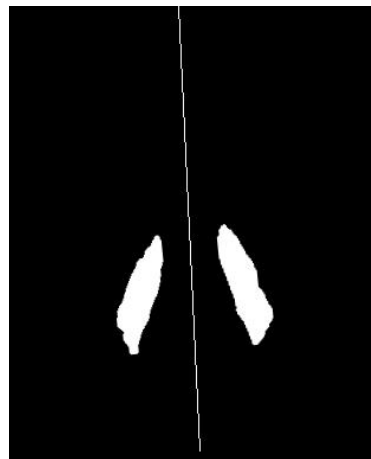


(A) To the left of

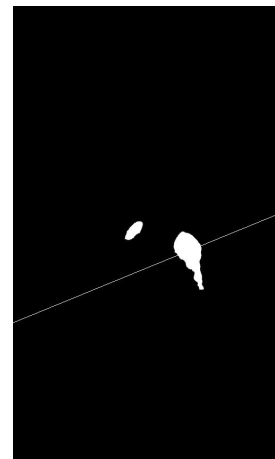


(B) Completely to the left of

FIGURE 9.12: Comparison of the fuzzy landscapes corresponding to the relation *to the left of the liver* (figure 9.12A) and the relation *completely to the left of the liver* (figure 9.12B).



(A) The line of symmetry between both psoas muscles is displayed. It corresponds to a symmetry measure of 0.94.



(B) Example where the line of symmetry intersects with one organ. It corresponds to a symmetry measure of 0.80.

FIGURE 9.13: Examples of symmetry measure.

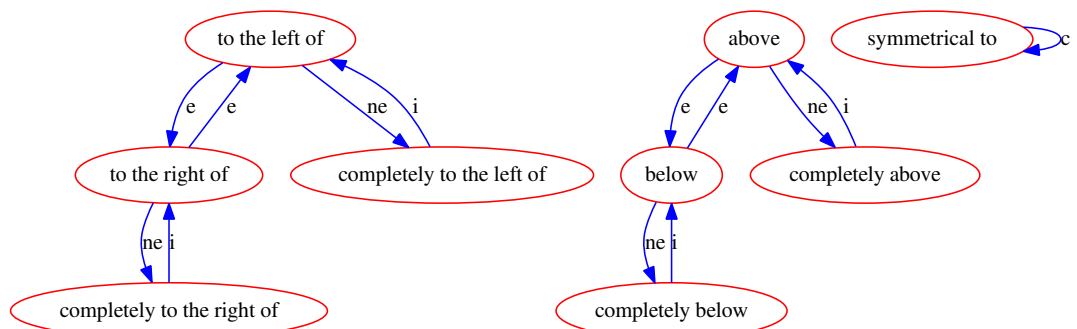


FIGURE 9.14: Graph representing the logical links between the relations handled by the model in this experiment.

We have 9 fuzzy dyadic relations and each instance contains 9 entities. Applied on all entities, that makes 640 relations to assess per instance. These relations and how they are logically linked are represented in Figure 9.14. For our second heuristic, that leads to the following order of evaluation:

$$\begin{aligned} & \text{symmetrical to} \rightarrow \text{above} \rightarrow \text{completely above} \rightarrow \text{below} \rightarrow \text{completely below} \rightarrow \\ & \text{to the left of} \rightarrow \text{to the right of} \rightarrow \text{completely to the left of} \rightarrow \text{completely to the right of} \end{aligned} \quad (9.4)$$

9.2.3 Workflow

The first two steps of the workflow are the same as in the experiment with the toy dataset in Section 9.1.4. In these steps, the only difference here is that entities are crisp. However, the process is exactly the same whether we deal with fuzzy or crisp objects.

As this is an annotation problem, step 3 is different. The goal is now to generate constraints for defining a FCSP. We know that the set of variables and the set of domains of this FCSP are:

$$V = \{v_{\text{liver}}, v_{\text{spleen}}, v_{\text{bladder}}, v_{\text{r_psoas}}, v_{\text{l_psoas}}, v_{\text{r_lung}}, v_{\text{l_lung}}, v_{\text{r_kidney}}, v_{\text{l_kidney}}\} \quad (9.5)$$

$$D = \{D_{\text{liver}}, D_{\text{spleen}}, D_{\text{bladder}}, D_{\text{r_psoas}}, D_{\text{l_psoas}}, D_{\text{r_lung}}, D_{\text{l_lung}}, D_{\text{r_kidney}}, D_{\text{l_kidney}}\} \quad (9.6)$$

where D_i is the set of all the possible values of v_i . The flexible constraints are generated from the frequent subsets of relations class by class as explained in section 6.2.3 on page 95. We get a set of constraints C . Furthermore, since every organ is unique, there cannot be identical labels in this problem. That means C has to be extended with constraints representing that two variables cannot be the same, which is the *AllDifferent* constraint:

$$\forall (v_i, v_j) \in V^2 \text{ such that } v_i \neq v_j, c_{i,j,\neq} = (v_i, v_j, R_{\neq}) \quad (9.7)$$

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

So the definition of the FCSP is made automatically. Then, once the FCSP is defined, for a given example, it can be solved as described in Section 6.2.2.3 on page 93 using the FAC-3 algorithm, for filtering inconsistent domain values, and the backtracking algorithm, for exploring the possible solutions. At the end, the entities of interest have been labeled and an explanation can be produced based on the constraints that were derived from the frequent itemsets.

9.2.4 Results

The model we build with our approach consists in the frequent subsets of relations that are extracted. There are as many hyperparameters as labels and they correspond to the thresholds used for assessing the frequency of a subset of relations. Model selection is necessary to get optimized thresholds. However, we have a small dataset so performing hyperparameter tuning by splitting the dataset into a training set, a validation set and a test set is not convenient. In order to get a better performance estimation and an efficient tuning of hyperparameters, we resorted to *nested cross-validation* (Cawley and Talbot, 2010): (1) an outer cross-validation is performed in which we get a training set and a test set for each iteration (this corresponds to a regular cross-validation), (2) an inner cross-validation is performed on the training set of the outer cross-validation to get an inner training set and a validation set for

tuning hyperparameters. This enables to get an unbiased tuning of hyperparameters while also having the advantages of cross-validation.

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

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

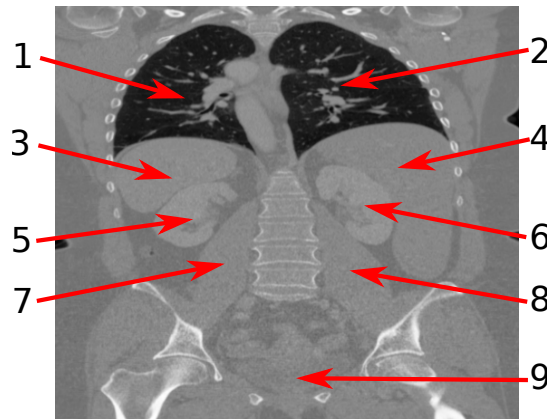
One example of explained annotations is displayed in Figure 9.15. The length of the explanation directly depends on the number of constraints in which the organs were involved. Thus, all organs were involved in at least 4 constraints, except the bladder. Only one constraint involving this organ was learnt, which may not be ideal for generating an explanation. This will be tackled in Section 9.2.5 where we evaluate explanations. We also notice that several reciprocal relations are present in explanations. This was expected based on our observations from the previous experiment (cf. Section 9.1). However, we get situations where two different direction accuracies are used: for example, in the explanation of entity 3 (the spleen) in Figure 9.15, we have:

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

One may expect to get both *completely below* and *completely above* or both *below* and *above* in the explanation. This is due to the fact that, because of their shapes, the fuzzy landscape corresponding to *completely below the left lung* covers a bigger area of the image than the fuzzy landscape corresponding to *completely above the spleen*. The constraints associated to the FCSP that enabled to annotate this instance are given in Appendix F.1 on page 183.

The average values for the minimum supports associated to each organ are displayed in Table 9.1. We can notice that several organs have a minimum support at 0.99, which means that several relations are almost always fully satisfied. This happens because these organs satisfy very well the directional relations in the vocabulary. For example, the relation *left lung to the left of right lung* is equal to 1 in every instance of the dataset. Thus, for such organs, even a high minimum support enables to extract relations that are relevant to the problem under study.

Table 9.1 also displays the average number of constraints per organ over all the folds of the cross-validation. In particular, we notice that the spleen is the organ that is represented in the greater number of constraints. Even though each organ satisfies relations in a different way, this observation is consistent with the fact that the spleen has the lowest minimum support by a large margin. However, We can also notice that right lung is involved on average in 8 constraints although its minimum support is equal to 0.99. This happens because it fully satisfies a few relations in every instance of the dataset, as described in the previous paragraph. Besides, there is always only one constraint involving the bladder, which is translated in the explanation of entity 9 in Figure 9.15. While we looked for the hyperparameters that enable to get the highest annotation performance, this may not always be compatible with the objective of explainability. This is one instance of the kind of tradeoff we encounter between performance and explainability.



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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

FIGURE 9.15: Example of explained annotations.

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

TABLE 9.1: Table representing several results for each class of organ when performing a nested cross-validation with 5 folds in the outer loop. First, the minimum support associated to the learning of the frequent subsets of relations is given for each class. In the second column, the average number of constraints that we get for each class of organ at the end of the learning is displayed. Finally, we show the average confidence that we got for each annotation during the testing (accuracy of 100%).

The last column in Table 9.1 presents the average values of the confidences associated to the annotations of each organ. We notice that the two lowest average confidences correspond to the two classes of organ that have the lowest minimum supports. This is logical since the minimum support is one of the two factor in the computation of the confidence. All the organs that have a high minimum support and, on average, several constraints get a high confidence, which confirms the relevance of the constraints that were extracted during the learning phase. However, the case of the bladder is again interesting. It gets a high confidence although it is linked to only one constraint that may not seem relevant to a human (cf. Figure 9.15. We discuss this in further detail in Section 9.2.5.

We also investigated the number of training examples needed for our model to perform well. Using the nested cross-validation, we can evaluate the performance of the model for a number of training examples ranging from 17 to 34. Then, doing a reverse cross-validation (the training set and the test set are inverted), we can assess the performance of the model for a number of training examples ranging from 1 to 17. In that situation, instances are part of the test set in several iterations, but we ensure that the model has learnt from all the possible combinations of instances.

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

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

- Fuzzy directional landscapes since they rely on the computation of a fuzzy

dilation (cf. Chapter 8). Since the evaluation of a relation is not expensive once the corresponding fuzzy landscape has been generated, we would like to avoid computing fuzzy landscapes if possible. For example, the fuzzy landscape corresponding to *to the right of the liver* is not generated if all the relations e to the right of the liver (e being any entity different from the liver) have been previously filtered by the heuristic. This is a strong constraint but in practice it is often satisfied when an entity is close to an edge of the image. For instance, if an entity e is always in the top left corner of the images, it is likely that the fuzzy landscapes *to the left of e* or *above e* will be dismissed at one point.

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

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

For the second heuristic, there are two different cases. First, the symmetry relation is commutative. That means that half of the evaluations of this relation can be saved. As mentioned in the previous paragraph, there was originally 2520 symmetries to evaluate and this number is thus reduced to 1260. For directional relations, the order presented in Equation (9.4) is followed to make the most of the logical links between these relations. It is important to note that once a relation is discarded with the first heuristic, we do not count it as a saved computation here. The results are presented in Table 9.2. This strategy enables to avoid computing the evaluation of 1636 relations, which represents 7.6% of the total number of evaluations. In particular, we notice that the logical link \xrightarrow{ne} is more efficient than \xrightarrow{e} on this dataset. This is due to the fact that there are more relations whose evaluation is equal to 0 than relations which are evaluated to 1: indeed, 3.8% of the evaluations

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

TABLE 9.2: Table presenting the number of prevented evaluations using our second heuristic (cf. section 5.3 on page 74). Following the topological sort that we got in section 9.2.2 on page 141, the results for each logical link are shown. Overall, this strategy enables to prevent 1636 evaluations, which represents 7.6% of the total number of evaluations.

are equal to 1 whereas about 50% of them are equal to 0. This high proportion of null evaluations also explains the efficiency of the first heuristic.

Overall, combining both heuristics, we manage to avoid computing about 40% of the total number of evaluations (8596 out of 21420 evaluations). In particular, it enables to avoid computing expensive relations like the symmetry or morphological directional relations. Coupled with the work presented in Chapter 8, it has led to a significant improvement in the execution time of the evaluation step. However, we notice that the first heuristic, which aims at pruning infrequent relations online, is more efficient by a large margin.

9.2.5 Evaluation of Explanations

We presented in Section 1.4 on page 32 the current state of the art of evaluation methods for XAI, which can be divided into three categories: application-grounded, human-grounded and functionally-grounded evaluations. Since the approach we propose in this thesis aims at providing an explainable classification or annotation model for different types of input signal and in different fields, we decided to perform a human-grounded evaluation that relies on the method proposed by (Baaj and Poli, 2019). It consists in assessing a set of assertions that evaluate the language used in the explanations, the content and the form of the explanations, and how helpful to humans they are. The full list of assertions is displayed in Table 1.2 on page 35.

In this experiment, we asked a panel of participants to answer the following assertions:

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

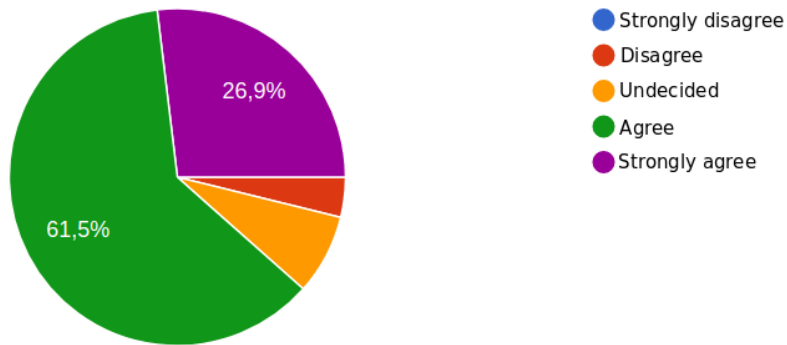
Participants were asked to assess these assertions using the following Likert scale (Likert, 1932):

- Strongly disagree,
- Disagree,
- Undecided,
- Agree,
- Strongly agree.

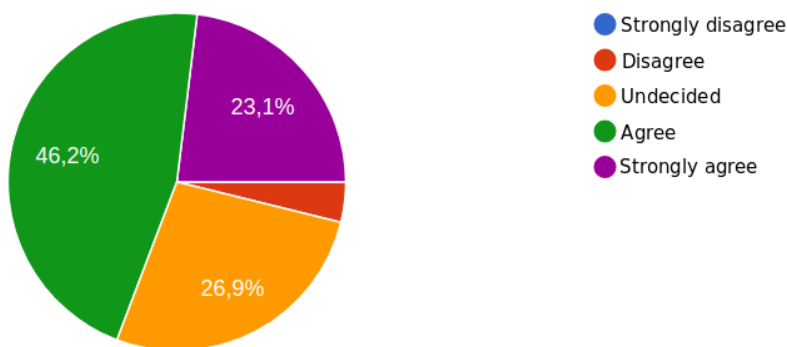
Participants also had the possibility to insert any comment at each evaluation.

The panel of participants was composed of 26 people. One of the participant was a medical doctor and the others were students and researchers from CEA and CentraleSupélec.

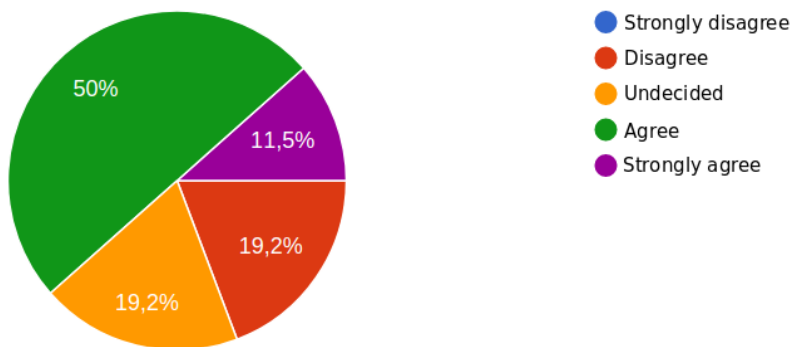
Explanations are simple and easy to read:



Explanations are convincing:



Data and explanations are sufficient to trust the system:



Explanations indirectly express the way the system reasons:

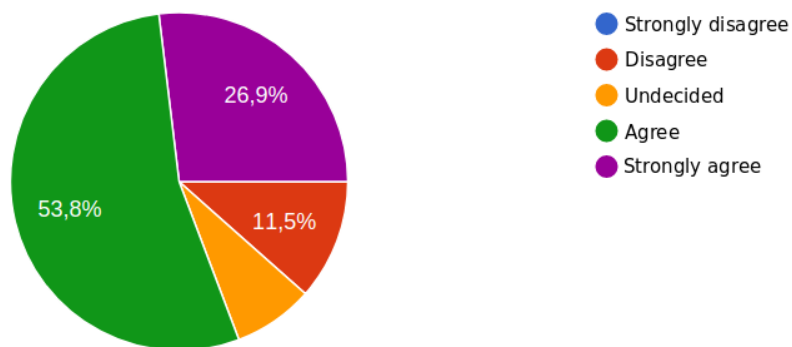
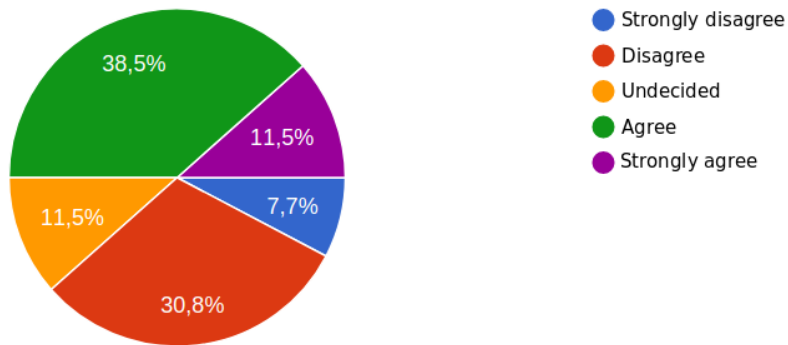
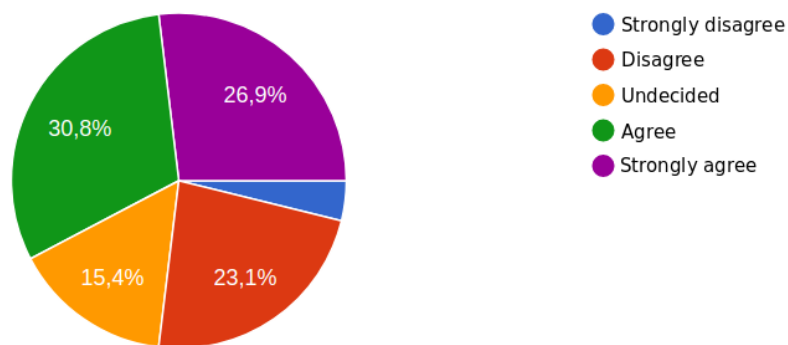


FIGURE 9.16: Answers to the first four assertions.

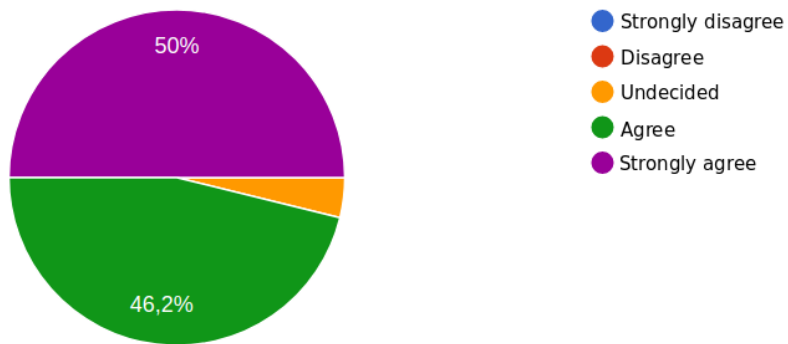
The length of the explanations is adequate:



It is difficult to read explanations until the end:



Explanations seem consistent:



Explanations are true:

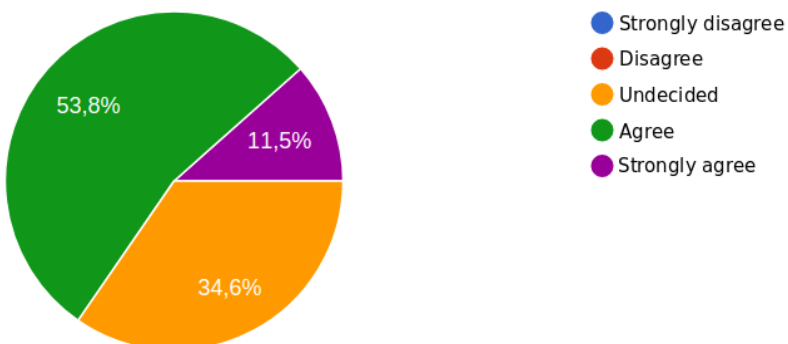


FIGURE 9.17: Answers to the last four assertions.

The results we got are displayed in Figure 9.16 and Figure 9.17. Regarding the form of the explanations, 88% of the participants think that explanations are simple and easy to read. However, about 30% of them believe that it is difficult to read explanations until the end and 15% are undecided. That suggests that explanations may be too long, which is supported by the assessment of the length of the explanations. Indeed, half of the participants are not convinced by the length of the explanations. Thus, this is a point to improve. In our approach, a higher minimum support should lead to shorter explanations but it also has a direct impact on the performance of the model. There is no clear way to favour one criterion over another.

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

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

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

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

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

9.3 Discussion

In this chapter, we presented two experiments we carried out to test our approach on two image datasets: an artificial one and a real one. These tests enable to validate the advantages of our approach and to spot a few areas of improvement.

The explanations produced by our model have been overall well graded by a pool of participants. Most of them agree that the explanations are convincing and enable to trust the system. This is important because explainability is the main goal of the approach we propose in this work. However, while our explanations are consistent, they may rely on relations that may not always seem relevant to the end-user. This could be solved by improving the vocabulary and by tweaking the learning

process. For instance, in the case of organ annotation, non-local relations could be penalized during the learning phase.

Regarding the performance of our model, it managed to reach a perfect accuracy in both cases when the segments corresponding to entities were already provided.

For preventing useless computations during the evaluation phase, we validated the performance of our online pruning heuristic. In each experiment, it enabled to prevent about 30% of the evaluations. Also, we saw in the explainable organ annotation experiment that it was well suited to dismiss infrequent expensive relations like symmetries and fuzzy directional relations. The second heuristic returned less impressive results when the number of logical links between relations is restricted.

The learning algorithm we use can generalize well from a small dataset if the instances are consistent with each other. This behaviour is an advantage because learning an explainable model on few data may not be easy depending on the type of model that is built. Besides, a few fields where explanations are needed usually involve small datasets, such as in medicine.

We also showed in our first experiment that our model can deal with fuzzy entities, which is an asset since entities may not always be precisely defined. This interesting property enables to combine our approach with a probabilistic or fuzzy segmentation algorithm which should help to take into account the vagueness in the input signals.

In the next chapter, we evaluate our approach on a problem of time series classification. It should be harder to handle since no segments of entities are provided and the vocabulary should be more limited since the literature about fuzzy temporal relations is thinner.

Chapter 10

Application to Time Series Classification

In this experiment, we aim at performing explainable time series classification. The time series we cope with come from (Hotel, 2017) and are multivariate. The challenges that we expect to face are the following:

- Unlike the previous experiments, we have to perform segmentation on the time series to obtain time windows in which we can compute interesting relations between the signals.
- The literature dealing with fuzzy temporal relations is much thinner than with fuzzy spatial relations, so the vocabulary should be relatively restricted. Therefore, the performance and the explanations of the model could be affected.

We first present the dataset we worked on. Then, we describe the vocabulary of relations we built for this experiment. The third section tackles the workflow of the approach. Finally, we present the results we obtained and discuss them.

10.1 Dataset

The original dataset contains 5 classes of multivariate time series. Each class represents a different toxic chemical. In order to characterize these chemicals, acquisitions have been made to obtain 8 electrical signals that represent the response of a sensor to a gas. These 8 responses form a chemical signature that should enable to classify the toxic chemicals.

We have normalized all the instances in the dataset so that all signals fit between 0 and 1. Figure 10.1 and Figure 10.2 display examples from all the classes. We notice that class 4 contains instances that have very different patterns from each other, as shown in Figure 10.2. This is an issue for our approach because it is based on the assumption that all or most of the instances from a given class should be highly correlated with each other (cf. Chapter 4). As a consequence, we decided to discard this class and to focus only on classes 1,2,3 and 5.

Since signals are noisy, we smoothed them using a Gaussian filter with a standard deviation equal to 8. Also, the signals have two distinct states: a *steady* state, during which the signal is stable in time, and a *transient* state, during which the signal evolves to reach the steady state. This is why we decided to perform segmentation on each time series. We opted for a multivariate time series segmentation algorithm called *Greedy Gaussian Segmentation* (Hallac et al., 2019). With this method, we can choose beforehand the number of segments we want. In our case, we decided to segment time series into two parts with one segment representing the transient state and the other one representing the steady state. An example of the resulting time

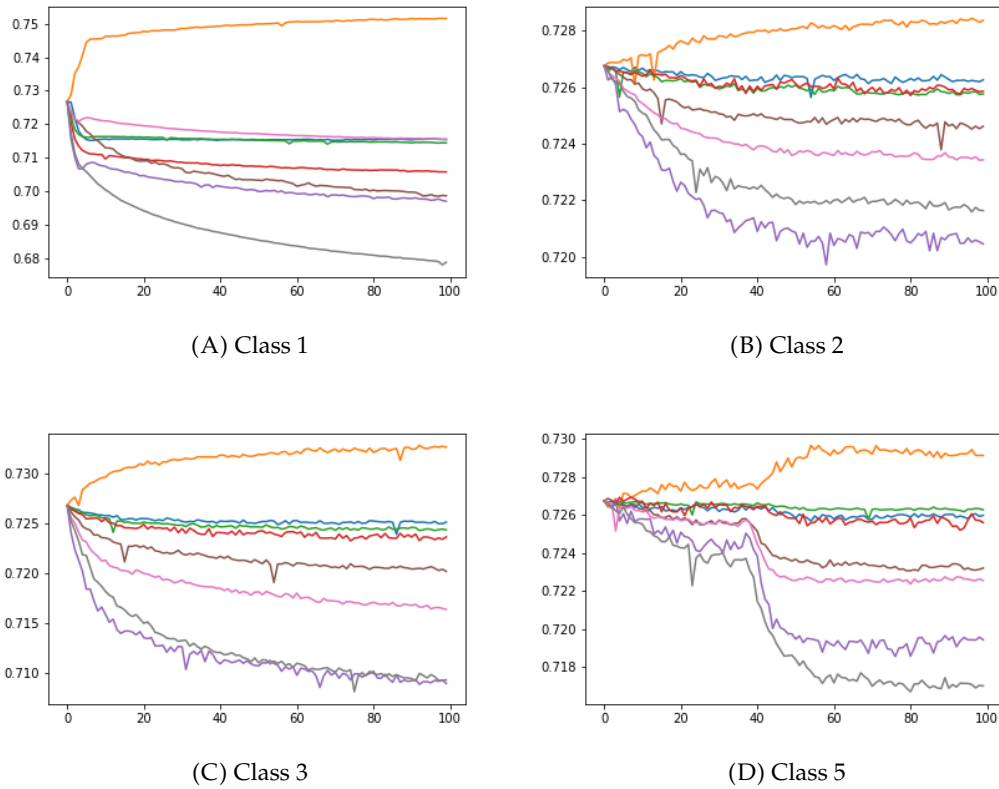


FIGURE 10.1: Examples from the time series dataset we worked on.

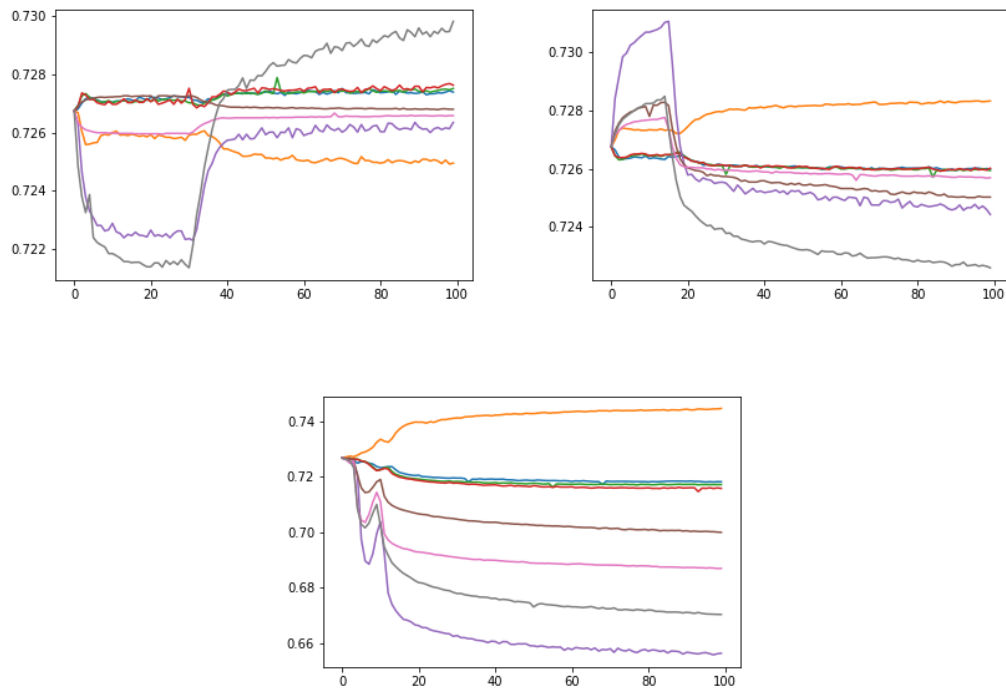


FIGURE 10.2: Three examples from class 4 that do not present the same properties at first sight. Those are the three types of patterns that we encounter in the instances of class 4.

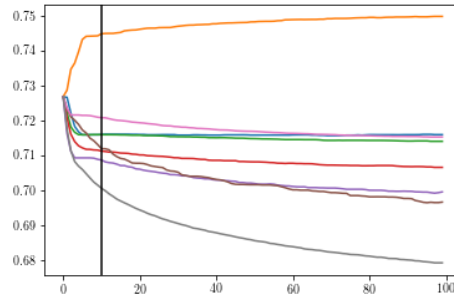


FIGURE 10.3: Example of a time series from class 1. The 8 signals have been smoothed and a segmentation has been performed. The vertical black line represents the border between the first segment, corresponding to transient state, and the second segment, corresponding to steady state.

series and their segments is shown in Figure 10.3. Moreover, since transient state is not representative of the final state of the signals, which determines the class of the time series, we decided to focus only on the steady state of each time series.

Thus, we work on a dataset of multivariate time series that have been smoothed and segmented. It contains 606 images distributed as follows: 170 instances from class 1, 138 instances from class 2, 158 instances from class 3 and 140 instances from class 5.

In the following, we will refer to the different signals in the instances by their color (signal blue, signal red,...).

10.2 Vocabulary of Relations

In this experiment, we resorted to 11 unary relations and 7 dyadic relations. The unary relations aim at characterizing the evolution of the signal. Among them, a first type of relations enables us to assess the *variations* of a signal over a given time scope. Given an instance in the dataset, we first compute the standard deviation of each signal over the steady state. Then, we define a linguistic variable *Varies* associated to the following fuzzy sets defined over the space of real numbers: *varies very little*, *varies little*, *varies averagely*, *varies much* and *varies very much*. These five different fuzzy sets are displayed in Appendix F.2 on page 184.

The two other types of unary relations in the vocabulary assess the *increase* or the *decrease* of a signal over a time scope. It relies on computing the gradient, which enables to get information about the direction in which a signal is evolving. Two linguistic variables *Increases* and *Decreases* are specified. Each one of these variables is associated to three fuzzy sets such as: *decreases little*, *decreases averagely*, *decreases much*, *increases little*, *increases averagely* and *increases much*. These fuzzy sets are also displayed in Appendix F.2 on page 184.

For dyadic relations, we defined two relations that compare if a signal is *greater* or *lower* than another over a time scope. In order to assess these two relations, we first compute the difference $s_1 - s_2$ between a signal s_1 and a signal s_2 . Then, we can compute the proportion of time when the difference is positive (greater) or negative (lower) over the whole time scope. This enables to get the relations *greater than* and *lower than*.

The last type of relations, which characterizes the *distance* between two signals, is also based on the difference between s_1 and s_2 . We compute the mean of the absolute value of this difference. Then, we define a linguistic variable *Distance* that is associated to 5 different fuzzy sets: *very close to*, *close to*, *at an average distance from*, *far from* and *very far from*. They are displayed in Appendix F.2 on page 184.

Overall, we have 11 unary relations and 7 dyadic relations. On 606 instances, computing relations only during the steady state leads to 290880 evaluations.

The graph displayed in Figure 10.4 shows the logical links between these relations. Thus, we get the following order on relations:

lower than \rightarrow greater than \rightarrow very far from \rightarrow close to \rightarrow
 at an average distance from \rightarrow very close to \rightarrow far from \rightarrow varies very much \rightarrow
 varies little \rightarrow varies very little \rightarrow varies much \rightarrow varies averagely \rightarrow decrease little \rightarrow
 increase averagely \rightarrow increase little \rightarrow increase much \rightarrow decrease much \rightarrow
 decrease averagely

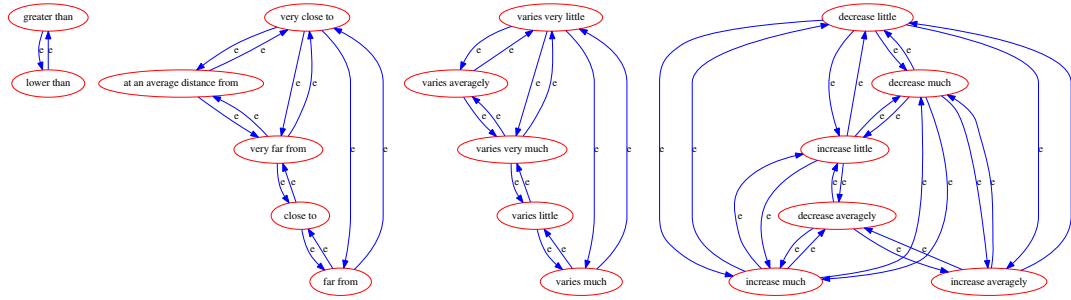


FIGURE 10.4: Graph representing the logical links between relations for the time series classification experiment.

10.2.1 Workflow

In this experiment, the workflow is almost the same as for the toy dataset (cf. Section 9.1.4). The only difference is that, in the first step, each instance has to be segmented before evaluating relations. This is important because the quality of the segmentation could have an impact on the performance of the model, even though fuzzy logic enables to deal with relatively imprecise segments.

10.2.2 Results

Four examples of explained classifications are displayed in Figures 10.5 to 10.8 (one for each class). In the following paragraphs, we interpret the quantitative results we obtained, which enable to understand these examples.

We tested our model doing a 10-fold stratified cross-validation. For hyperparameter tuning, since the dataset is big enough, we performed an hold-out validation with a validation set using the same Bayesian optimization method as in Section 9.2.4. The hyperparameters that we used are displayed in Table 10.1. The results we got at the end of the cross-validation are shown in Figure 10.9 and we obtained an accuracy of 81%. Classes 1,2 and 3 are relatively well classified but the performance of the model drops for class 5. We see in Table 10.1 that the rules from class

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

- the blue signal *increases little* in steady state,
- the purple signal is *greater than* the grey signal in steady state.

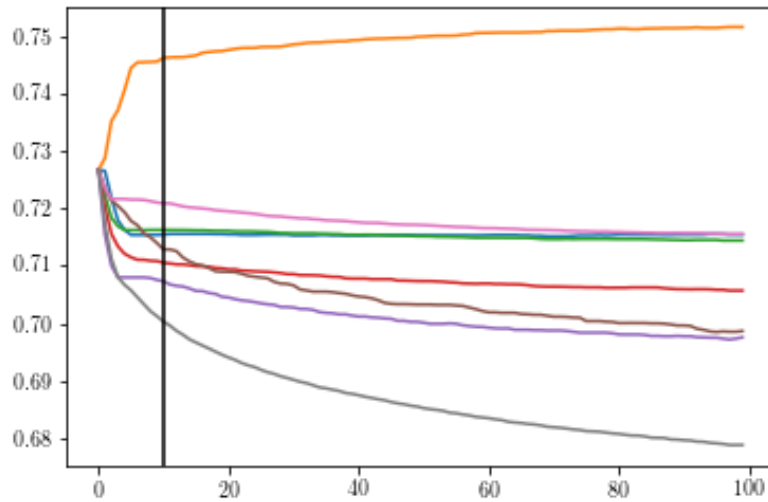


FIGURE 10.5: Example of explanation for an instance from class 1.

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

- the pink signal is *at an average distance from* the brown signal in steady state,
- the brown signal is *greater than* the pink signal in steady state,
- the grey signal is *greater than* the purple signal in steady state.

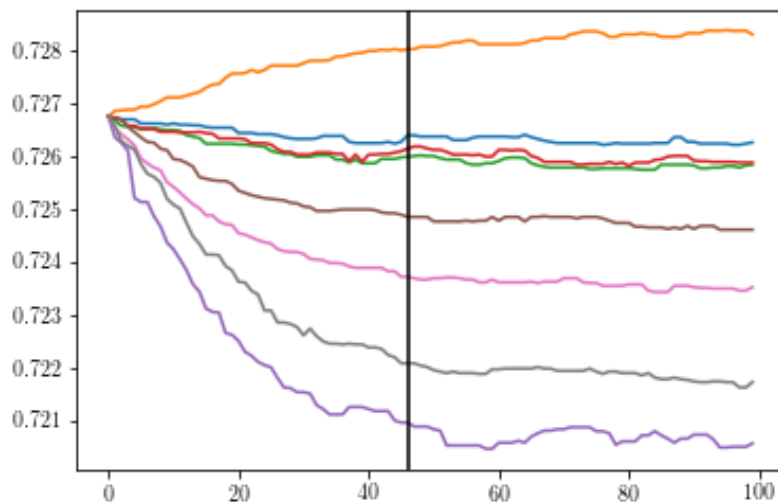


FIGURE 10.6: Example of explanation for an instance from class 2.

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

- the pink signal *varies much* in steady state,
- the red signal is *close to* the blue signal in steady state.

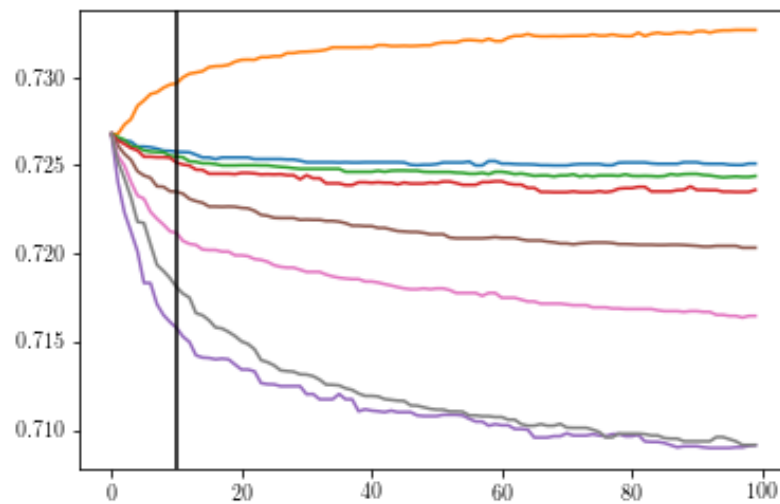


FIGURE 10.7: Example of explanation for an instance from class 3.

This instance belongs to class 5 with a high confidence because:

- the blue signal *varies little* in steady state.

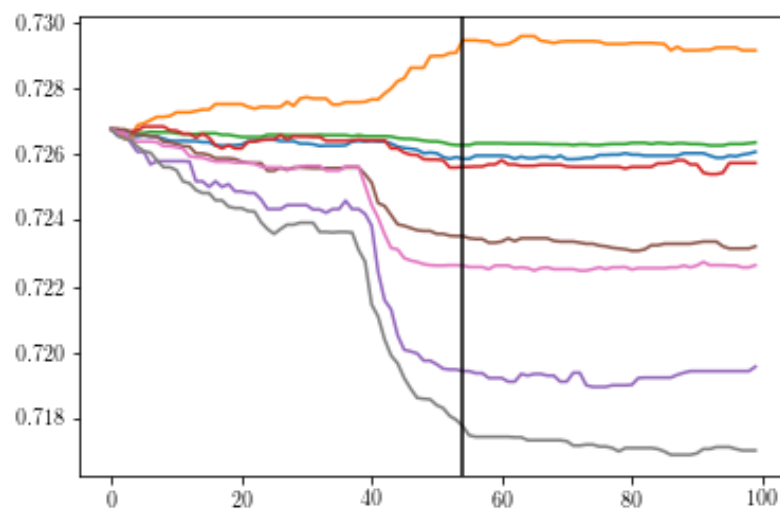


FIGURE 10.8: Example of explanation for an instance from class 5.

Class	Minimum support	Average number of relations in rules before pruning	Average number of relations in rules after pruning
1	0.89	72.9	2
2	0.85	90.1	3
3	0.87	97.2	1.9
5	0.88	91.5	1

TABLE 10.1: The second column shows the values of minimum support we found for each class after tuning. The third column shows the average length of rules before the relations that are shared by rules from other classes are removed. The last column displays the average length of rules after they have been pruned.

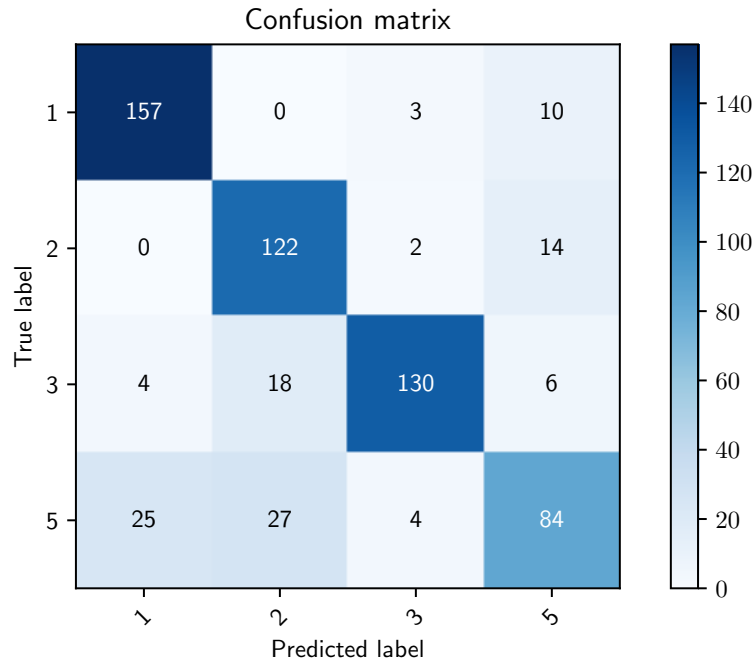


FIGURE 10.9: Confusion matrix displaying the performance of our model in the time series classification experiment.

5 have only one relation left after pruning. This is actually only one rule with a single relation in its antecedent: *the blue signal varies little in steady state* (cf. Figure 10.8). While this relation is a valid element of description of class 5, the same relation could be used to describe other classes, such as class 2 for example. The fact that no dyadic relation is still part of the rules after pruning may harm the classification. This is a situation similar to the one we encountered with the bladder in Section 9.2, i.e. one relation has been extracted but the explanation it entails is not convincing enough. This may indicate that our vocabulary is not rich enough to classify class 5 well.

Table 10.1 also shows the average number of relations in rules before and after pruning for each class. We notice that at least 97% of the relations are pruned. On one hand, it enables to get short rules that thus lead to short explanations, which is positive in light of the results of the survey we presented in the previous experiment. On the other hand, this is a clue that the vocabulary leads to descriptive but not

discriminative relations. This may indicate that the vocabulary of relations was not set well enough.

In order to verify this point, we trained a random forest classifier on the set of evaluated relations we get at the end of the evaluation phase. It is a forest of 10 trees trained using Gini criterion for splitting. Performing the same cross-validation, we get an accuracy of 99% on the whole dataset, which may validate the vocabulary of relations and their evaluations. While we should not expect the same performance of classification rules, this shows that our rule learning algorithm is not as well suited to this kind of data as some other learning algorithms. One possible explanation to this phenomenon is that the random forest classifier learns more complex decision bounds in the feature space than our model. While it enables to get a much better performance, it may not help to explain the results. Indeed, when computing the feature importance of the random forest trained on the whole training set, the most important feature does not account for more than 4.5% of these. This is much less than the relative proportions of the four classes (28%, 23%, 26% and 23% respectively), which may show that no relation is truly representative of any class. Thus, the random forest can perform a more complex reasoning in order to combine features effectively whereas our rules are short after pruning (3 relations at most on average). Such short rules cannot reach the same level of performance in the absence of discriminative features.

Also, we noticed that, for several classifications, we had similar high confidences in predicting two different classes for the same instance. So, for several instances, rules struggle to discriminate between two classes. The explanations are useful to understand this behaviour: looking at Figure 10.5 and Figure 10.8, one explanation could fit the other instance and vice versa. So, while the results are not as good as we expected, this use case shows that explanations are important to understand what the model does and where its mistakes come from.

Regarding our goal to avoid computing useless relations, we also evaluated our two heuristics on this dataset. It should be noted that the relations in the vocabulary are not expensive, which is logical since we are dealing with 1D signals, and that preventing useless evaluations is not as important in this experiment as in the previous two.

With our online pruning heuristic, we prevent the computation of 98023 evaluations out of 290880. That represents about 34% of the total number of evaluations to perform, which is the same magnitude as in the previous experiments.

As shown in Figure 10.4, there are many logical links between the relations in our vocabulary in this experiment. That is why we are able to prevent about 20% of the evaluations. This good performance is also due to the fact that there are many relations that are fully satisfied, which enables to make the most of the logical links existing between the relations.

Overall, we are able to save about 54% of the total number of evaluations, which is a very good result.

10.3 Discussion

In this chapter, we stressed the genericity and limitations of our approach by its application to a new use case : the classification of time series in order to recognize several classes of toxic chemicals. We first showed that our approach is able to deal with a raw segmentation of the inputs. This will enable to deal with a wider variety

of tasks. We also had to manage a different type of vocabulary since fuzzy temporal relations are less common than spatial ones. Our model did not reach the same level of performance as a random forest classifier. This highlights the interest of explanations in a different situation. Indeed, unlike the random forest, our model enables to understand why an instance has been classified well or misclassified.

One solution to improve the performance of our approach consists in expressing more complex rules. The current model handles conjunctions of relations and thus may be enhanced with other operators, such as disjunctions or negations. It also highlights the limits of our frequency-based learning approach. Some of these prospects are discussed in Section 10.3.

Conclusion and Perspectives

In this thesis, the goal was to make a transparent model that would be able to provide explanations in natural language for the decisions it makes. Thus, we focused on proposing an XAI approach that aims at building such a model for solving classification and annotation tasks.

Also, we wanted our model is based on explicit expressive relations due to their known importance in human understanding. Relying on a vocabulary defined by an expert, we investigate potentially relevant and interpretable relations. This strategy enables to avoid building a dataset where relations are annotated for each instance, which is expensive to get.

Given a training set and a vocabulary of fuzzy relations, our approach first assesses the different relations in instances according to the vocabulary. Then, a fuzzy frequent itemset mining method is applied to extract subsets of frequent fuzzy relations. Rules, for classification, or constraints, since annotation relies on a FCSP, can then be built. After having performed the desired task, the model produces an explanation based on the relations that contributed to the decision. Since these relations are associated to a linguistic description, we get an explanation in natural language.

In the following, we first discuss our main contributions and then present several directions for further research.

Contributions

Learning Relations

To prevent the need of relation annotations, which is costly, we built our approach on class annotations. We then developed a strategy whose goal is to extract from training data the most frequent patterns in a given set of computed relations. This idea is motivated by the observation that instances belonging to the same class should contain a few relations that are representative of this class. Thus, by extracting the subsets of relations that are frequently observed in training instances, we should get a description of each class that can be used for performing classification or annotation.

As a consequence, we decided to focus on frequent itemset mining to extract the most frequent subset of relations for each class. The Close algorithm (Pasquier et al., 1999) is well suited for this task since it performs well on correlated data. However, it is unable to deal with fuzzy data.

We proposed a new algorithm, *fuzzy Close*, which relies on a closure operator that enables to deal with a formal fuzzy context. We demonstrated that this new operator is actually a closure one and that it can extend the original method to cope with fuzzy data.

We showed on two images datasets that it can learn relevant relations for the problem to solve. The performance of the algorithm highly depends on the value of

its hyperparameter, the minimum support, that enables to assess if a subset is frequent or not. If this hyperparameter is too high, extracted relations are too generic and not discriminative enough to achieve good performance. This is a case of underfitting. If its value is too low, irrelevant relations will be extracted. That would harm the generalization ability of the model and leads to overfitting.

We also tested it on an application of time series classification. While the vocabulary of relations seems to be convenient, the approach extracts frequent subsets of relations that are then expressed as conjunctions of relations. This leads to two constraints:

- Only conjunctions of relations are expressed. Other operation such as disjunction or existence could certainly improve the performance of the model and would not be harder to integrate in an explanation.
- Only frequent subsets of relations are extracted. That means the model learns the relations that are frequently present together. However, it does not take into account relations that are not or barely satisfied. Those relations could also enable to get a richer model and better explanations.

Heuristics for Preventing Useless Evaluations

The first step of our approach consists in evaluating relations from a vocabulary between all the possible combinations of entities in the instances of the training set. Thus, the total number of evaluations to perform grows quickly with the number of relations and their arities, with the number of entities in instances and with the number of instances.

In order to avoid computing all these evaluations, we develop two heuristics. The first one is based on the following principle: if a relation is (almost) never satisfied in the instances that have been evaluated, it may increase our confidence that this relation will not be frequent at the end of the evaluation process. In particular, we made the assumption that the minimum support should be at least 0.5 to have representative subsets of relations. Therefore, as soon as a relation is bound to be infrequent, it is discarded from the set of relations to evaluate. We showed in our experiments that we can prevent about 30% of the evaluations, including some evaluations relying on relations that are expensive to compute.

The second heuristic aims at propagating the results of one relation to another when it is possible. It relies on three kinds of links between relations that can be obtained from their definition: the dependence between two relations, the logical implication between two relations and/or their complements, and the symmetry of a relation. We can represent these links in a graph. We presented an algorithm that converts this graph in a directed acyclic graph so that a topological sort can be applied. Thus, we get an order of evaluation on relations that enables to propagate the results of some evaluations based on the nature of the links between the relations to evaluate. This heuristic strongly depends on the number of relations in the vocabulary that satisfy such links, and on whether there are many fully satisfied or null relations. This is why we got better results when more relations are fully satisfied, such as in the experiment on time series classification.

Fast Parallel SIMD-based Fuzzy Dilation Operator

To make the evaluation process faster, we first relied on heuristics to avoid unnecessary computations. When computations have to be made, another area of improvement is to ensure that we evaluate relations as fast as possible. On images, several relations rely on a fuzzy dilation, which is a very expensive operator. The first area of improvement was to avoid considering pixels that do not contribute to the final result. This tweak enables to prevent a number of operations proportional to the ratio between the size of the image and the size of the object under consideration.

Also, this operator is well suited to parallel computing, which is why we worked on a multithreaded implementation that uses SIMD explicit instructions. We showed that this implementation enables to top the state-of-the-art algorithm in several situations, including on the dataset of medical images we used in our experiments.

Besides, this implementation can be extended to the 3D case, which is convenient for dealing with medical images.

Prospects

Building Richer Rules and Sets of Constraints

As we already mentioned, our fuzzy frequent itemset mining approach returns frequent subsets of relations that represent conjunctions of relations. In order to get more generic rules or sets of constraints, disjunctions could be considered. This would enable to take into account various representative subsets of relations better.

Also, the current strategy looks for relations that are frequently satisfied in the training set. However, the fact that a relation is rarely or never satisfied can also be a valuable piece of information. Thus, we would like to extract frequent subsets of relations where each relation is either (almost) fully satisfied or null in the instances of the training set. To achieve that, a new operator, based on the closure operator we currently use, could be defined so that, given an itemset I , it also tracks the relations that are always null when the relations in I are satisfied.

Another type of relations that would be interesting is *there exists*. In particular, when the number of entities varies from one instance to another, this would characterize whether another relation has been satisfied or not in instances. For example, if the instances from a given class always contain at least one square, this would be more generic than having one relation *is square* per entity.

Adjusting the Length and Scope of Explanations

In the evaluation of explanations, we saw that the length of the explanations was an issue for some participants. Also, the fact that some extracted relations are not local enough was pointed out. We could investigate a new learning process in which the goal is to optimize a loss function that directly depends on the length of the explanations (like in (Lakkaraju et al., 2016)) and the locality of the relations they contain. Thus, we would have more control on those properties.

Dealing with Unsupervised Segmentation

In the case of images, we worked with datasets where the segments of the entities to annotate were given. However, this may not be the case in another dataset. While a segmentation algorithm may be learnt, we may have to resort to unsupervised

segmentation. In that case, we will first get an oversegmentation of the image, as shown in Figure 10.10. To obtain segments that are closer to the entities we are working on, one idea is to apply a hierarchical clustering. An illustration of this approach is given in Figure 10.10.

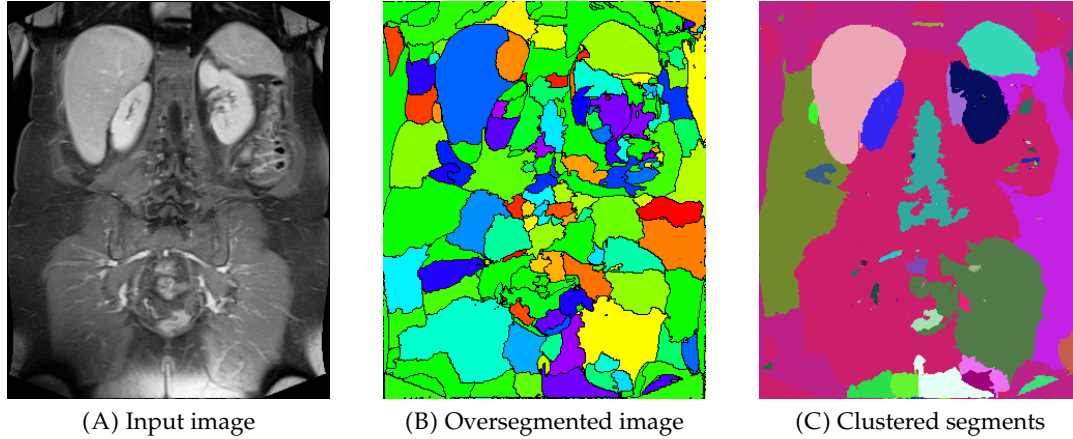


FIGURE 10.10: Given an input image, we can get a convenient segmentation of the entities of interest by applying hierarchical clustering on an oversegmentation.

Learning New Relations

Another area to explore is to not only assess relations on the training set, but also learn new relations from training instances. For example, parametric relations, such as directions or distances, could be adjusted using histograms of angles or distances.

Furthermore, new dilation-based relations could be defined by learning on the training set new structuring elements.

Another possibility is to use logic auto-encoders (Dumancic et al., 2019c). Their principle is the same as traditional auto-encoders except that they handle predicates and constants. In our case, the latent predicates that are learnt could be more generic relations that are relevant to the target task. Also, this could enable to get a shorter set of constraints, which would make the annotation faster.

Evaluating Explanations

Explanation evaluation is one of the biggest current issue in XAI. While a few methods have been proposed, it is very difficult to compare two different explainable models. As a consequence, expert assessment or surveys are usually performed. However, at the scale of one specific application, it may be possible to benchmark different approaches efficiently.

Some criteria used to assess an explanation could also be automatically evaluated. For example, in the case of image annotation and for assessing the consistency of an explanation, the principle could be the following:

1. computing all the fuzzy landscapes involved in the explanation,
2. computing the intersection of these fuzzy landscapes,
3. comparing this intersection with the ground truth to assess if the explanation is consistent.

While a few criteria will always require human judgment, the development of a few methods to perform an automatic evaluation of the explanation would provide a huge boost to the field of XAI.

Appendix A

Publications

A.1 International Peer-Reviewed Conferences

- **Spatial Relation Learning for Explainable Image Classification and Annotation in Critical Applications,**
Régis Pierrard, Jean-Philippe Poli and Céline Hudelot,
Artificial Intelligence, 2021.
- **SIMD-based Exact Parallel Fuzzy Dilation Operator for Fast Computing of Fuzzy Spatial Relations,**
Régis Pierrard, Laurent Cabaret, Jean-Philippe Poli and Céline Hudelot,
PPoPP Workshop on Programming Models for SIMD/Vector Processing, 2020.
- **A New Approach for Explainable Multiple Organ Annotation with Few Data,**
Régis Pierrard, Jean-Philippe Poli and Céline Hudelot,
IJCAI Workshop on Explainable Artificial Intelligence (XAI), 2019.
- **Learning Fuzzy Relations and Properties for Explainable Artificial Intelligence,**
Régis Pierrard, Jean-Philippe Poli and Céline Hudelot,
IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2018.
- **A Fuzzy Close Algorithm for Mining Fuzzy Association Rules,**
Régis Pierrard, Jean-Philippe Poli and Céline Hudelot,
International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU), 2018.

A.2 National Peer-Reviewed Conferences

- **Apprentissage de relations floues pour l'annotation sémantique expliquée avec peu de données,**
Régis Pierrard, Jean-Philippe Poli and Céline Hudelot,
Rencontres des Jeunes Chercheurs en Intelligence Artificielle, 2019.

Appendix B

Fuzzy Logic : Main Definitions

This chapter aims at gathering all the notions from the fuzzy set theory and fuzzy logic (Zadeh, 1965) that are used in this thesis.

Fuzzy logic can be seen as an extension of Boolean logic that enables to manage imprecision. While a value is either true or false in Boolean logic, it can range from 0 (false) to 1 (true) in fuzzy logic.

B.1 Fuzzy Set

B.1.1 Definition

Definition 35: Fuzzy Set

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

If F is a non-fuzzy set, $\mu_F(u)$ is either 0, i.e. u is not a member of F , or 1, i.e. u is a member of F .

B.1.2 Properties

In this subsection, we define the *core*, the *support* and the α -*cut* of a fuzzy set.

Definition 36: Core of a Fuzzy Set

The *core* of a fuzzy set F defined on a universe \mathcal{U} is a non-fuzzy set defined as

$$\text{core}(F) = \{u \in \mathcal{U} | \mu_F(u) = 1\} . \quad (\text{B.1})$$

Definition 37: Support of a Fuzzy Set

The *support* of a fuzzy set F defined on a universe \mathcal{U} is a non-fuzzy set defined as

$$\text{supp}(F) = \{u \in \mathcal{U} | \mu_F(u) > 0\} . \quad (\text{B.2})$$

Definition 38: α -cut of a Fuzzy Set

For $\alpha \in [0; 1]$, the α -*cut* of a fuzzy set F defined on a universe \mathcal{U} is a non-fuzzy set defined as

$$F_\alpha = \{u \in \mathcal{U} | \mu_F(u) \geq \alpha\} . \quad (\text{B.3})$$

B.2 Linguistic Variable

Definition 39: Linguistic Variable

A linguistic variable (Zadeh, 1975) is defined as a triplet (V, Δ_V, F_V) such as:

- V is the name of the variable,
- Δ_V is the domain on which V is defined,
- $F_V = \{F_1, F_2, \dots\}$ is a finite collection of fuzzy sets. Each of these fuzzy sets is associated to a linguistic term which qualifies V .

B.3 Fuzzy Relations

Definition 40: Fuzzy Relation

Given two universes \mathcal{U} and \mathcal{W} , a binary fuzzy relation \mathcal{R} is characterized by a mapping defined as

$$\mu_{\mathcal{R}} : \mathcal{U} \times \mathcal{W} \rightarrow [0, 1] . \quad (\text{B.4})$$

It assigns a degree of relationship to any $(u, w) \in \mathcal{U} \times \mathcal{W}$. n -ary fuzzy relations are defined identically.

B.4 Fuzzy Operators

B.4.1 t -norm and t -conorm

Definition 41: t -norm

A t -norm is a function $t: [0; 1] \times [0; 1] \rightarrow [0; 1]$ which satisfies for all a, b, c, d in $[0; 1]$:

- $t(a, b) = t(b, a)$ (commutativity),
- $t(a, t(b, c)) = t(t(a, b), c)$ (associativity),
- $t(a, b) \leq t(c, d)$ if $a \leq c$ and $b \leq d$ (monotonicity),
- $t(a, 1) = a$ (neutral element).

Any t -norm is a fuzzy intersection operator. The most common one is Zadeh's t -norm, where $t = \min$.

Definition 42: *t*-conorm

A *t*-conorm is a function $T: [0;1] \times [0;1] \rightarrow [0;1]$ which satisfies for all a, b, c, d in $[0;1]$:

- $T(a, b) = T(b, a)$ (commutativity),
- $T(a, T(b, c)) = T(T(a, b), c)$ (associativity),
- $T(a, b) \leq T(c, d)$ if $a \leq c$ and $b \leq d$ (monotonicity),
- $T(a, 0) = a$ (neutral element).

Any *t*-conorm is a fuzzy union operator. The most common one is Zadeh's *t*-conorm, where $T = \max$.

B.4.2 Fuzzy Inclusion**Definition 43: Fuzzy Inclusion**

Let A and B be two fuzzy sets defined on a universe \mathcal{U} with the membership functions μ_A and μ_B respectively.

A is included in B ($A \subseteq B$), if, and only if, $\forall x \in \mathcal{U}, \mu_A(x) \leq \mu_B(x)$.

B.4.3 Fuzzy Closure Operator**Definition 44: Fuzzy Closure Operator**

Let $F_{\mathcal{U}}$ be a fuzzy set defined on a universe \mathcal{U} . A *fuzzy closure operator* h over $F_{\mathcal{U}}$ is defined as $h: F_{\mathcal{U}} \rightarrow F_{\mathcal{U}}$ and satisfies the following conditions:

- $\forall I \subseteq F_{\mathcal{U}}, I \subseteq h(I)$,
- $\forall I \subseteq F_{\mathcal{U}}, h(h(I)) = h(I)$,
- $\forall I, J \subseteq F_{\mathcal{U}}, I \subseteq J \Rightarrow h(I) \subseteq h(J)$.

B.4.4 Fuzzy Implication**Definition 45: Fuzzy Implication**

For any left-continuous *t*-norm t , there exists a unique function $\rightarrow: [0;1] \rightarrow [0;1]$ such that

$$\forall a, b, c \in [0;1], t(c, a) \leq b \text{ if, and only if, } c \leq (a \rightarrow b) \quad (\text{B.5})$$

In this work, we used the Łukasiewicz implication:

Definition 46: Łukasiewicz Implication

The Łukasiewicz implication \xrightarrow{L} is defined as

$$\begin{aligned} \xrightarrow{L}: [0;1] \times [0;1] &\rightarrow [0;1] \\ (a, b) &\mapsto \min(1 - a + b, 1) \end{aligned} \quad (\text{B.6})$$

B.4.5 Subsethood Degree

We define the *subsethood degree* that is used to define a fuzzy Galois connection.

Definition 47: Subsethood Degree (Höhle, 1996)

Let \mathcal{U} be a universe and $F^{\mathcal{U}}$ the set of all the fuzzy sets in the universe \mathcal{U} . The *subsethood degree* is defined as

$$\begin{aligned} \text{Subs}: F^{\mathcal{U}} \times F^{\mathcal{U}} &\rightarrow [0; 1] \\ (A, B) &\mapsto \inf_{u \in \mathcal{U}} (\mu_A(u) \rightarrow \mu_B(u)) \end{aligned} \quad (\text{B.7})$$

B.4.6 Fuzzy Complement

Definition 48: Fuzzy Complement

Let F be a fuzzy set defined on a universe \mathcal{U} and associated to the membership function μ_F . The fuzzy complement of F is defined by a membership function $\mu_{\bar{F}}$ such that $\forall u \in \mathcal{U}, \mu_{\bar{F}}(u) = c(\mu_F(u))$ with $c: [0; 1] \rightarrow [0; 1]$ a function that verifies:

- $c(0) = 1$ and $c(1) = 0$,
- $\forall z_1, z_2 \in [0; 1]$, if $z_1 < z_2$, then $c(z_1) > c(z_2)$,
- c is a continuous function,
- $\forall z \in [0; 1], c(c(z)) = z$.

In this work, we always use the *standard complement*, which is defined as

$$\begin{aligned} c: [0; 1] &\rightarrow [0; 1] \\ z &\mapsto 1 - z \end{aligned} \quad (\text{B.8})$$

Appendix C

Closure Operator

This chapter is dedicated to demonstrate that the operator h presented in Section 4.4.2 on page 63 is a closure operator. The first section reminds the core definitions of closure operators and the second section presents the proof.

C.1 Definitions

We first define what a *partially ordered set* (or *poset*) is.

Definition 49: Partially Ordered Set

A *partial order* over a set A is a binary relation \leq_A which satisfies

- $\forall a \in A, a \leq_A a$ (reflexivity),
- $\forall a_1, a_2 \in A$, if $a_1 \leq_A a_2$ and $a_2 \leq_A a_1$, then $a_1 = a_2$ (antisymmetry),
- $\forall a_1, a_2, a_3 \in A$, if $a_1 \leq_A a_2$ and $a_2 \leq_A a_3$, then $a_1 \leq_A a_3$ (transitivity).

A set A with a *partial order* \leq_A is called a *partially ordered set* and is written (A, \leq_A) .

We can then define what a *closure operator* and a *closed itemset* are.

Definition 50: Closure Operator

A *closure operator* on a poset (A, \leq_A) is a function $h : A \rightarrow A$ such as

1. $\forall a \in A, a \leq_A h(a)$,
2. $\forall a_1, a_2 \in A, a_1 \leq_A a_2 \Rightarrow h(a_1) \leq_A h(a_2)$,
3. $\forall a \in A, h(a) = h \circ h(a)$.

Definition 51: Closed Itemset

Let h be a closure operator over $(\mathcal{I}, \leq_{\mathcal{I}})$. An itemset $I \subseteq \mathcal{I}$ is said to be *closed* if, and only if, $h(I) = I$.

C.2 Proof

Let \mathcal{I} be a set of items, \mathcal{T} a set of transactions and \mathcal{R} a dyadic fuzzy relation such as $\langle \mathcal{T}, \mathcal{I}, \mathcal{R} \rangle$ forms a fuzzy formal context.

The proof we propose is the following:

Proof.

- h satisfies the first condition in Definition 50 because:

$$\forall I \in \mathcal{P}(\mathcal{I}), \forall i \in I,$$

$$\mu_I(i) = 1 \Rightarrow \forall t \in \mathcal{T}, \mu_{I\downarrow}(t) = \inf_{y \in \mathcal{I}} [\min(1, 1 + \mathcal{R}(t, y) - \mu_I(y))] = \min_{y \in I} \mathcal{R}(t, y)$$

$$\forall i \in I,$$

$$\mu_{I\uparrow\downarrow}(i) = \min_{t \in \mathcal{T}} [\min(1, 1 + \mathcal{R}(t, i) - \min_{y \in I} \mathcal{R}(t, y))] = 1$$

$$\text{So } h(i) = 1 \text{ and } i \in h(I)$$

$$\text{Thus } I \subset h(I).$$

- h satisfies the second condition in Definition 50 because:

$$\text{Let } I, J \in \mathcal{P}(\mathcal{I}) \text{ such as } I \subset J.$$

$$\forall i \in \mathcal{I}, \mu_I(i) \leq \mu_J(i)$$

$$\text{So } \forall t \in \mathcal{T},$$

$$\min_{i \in \mathcal{I}} [\min(1, 1 - \mu_I(i) + \mathcal{R}(t, i))] \geq \min_{i \in \mathcal{I}} [\min(1, 1 - \mu_J(i) + \mathcal{R}(t, i))]$$

$$\text{Thus } \mu_{I\downarrow}(t) \geq \mu_{J\downarrow}(t).$$

$$\forall i \in \mathcal{I},$$

$$\min_{t \in \mathcal{T}} [\min(1, 1 - \mu_{I\downarrow}(t) + \mathcal{R}(t, i))] \leq \min_{t \in \mathcal{T}} [\min(1, 1 - \mu_{J\downarrow}(t) + \mathcal{R}(t, i))]$$

$$\text{So } \mu_{I\uparrow\downarrow}(i) \leq \mu_{J\uparrow\downarrow}(i)$$

$$\text{Thus } \text{core}(I^{\uparrow\downarrow}) \subset \text{core}(J^{\uparrow\downarrow}) \text{ and } h(I) \subset h(J).$$

- h satisfies the third condition in Definition 50 because:

We already proved that $h(I) \subset h(h(I)) \forall I \in \mathcal{P}(\mathcal{I})$. Then, we have to show that

$$h(h(I)) \subset h(I).$$

$$\forall i \in h(h(I)), \mu_{h(I)\uparrow\downarrow}(i) = 1 \Rightarrow \mathcal{R}(t, i) \geq \mu_{h(I)\downarrow}(t), \forall t \in \mathcal{T}$$

$$\mu_{h(I)\downarrow}(t) = \inf_{y \in \mathcal{I}} [\min(1, 1 + \mathcal{R}(t, y) - \mu_{h(I)}(y))] = \min_{y \in h(I)} \mathcal{R}(t, y)$$

$$\text{So } \mathcal{R}(t, i) \geq \min_{y \in h(I)} \mathcal{R}(t, y), \forall t \in \mathcal{T}.$$

$$\text{Similarly, } \forall i \in h(I), \forall t \in \mathcal{T}, \mathcal{R}(t, i) \geq \min_{y \in I} \mathcal{R}(t, y)$$

$$\text{So } \min_{y \in h(I)} \mathcal{R}(t, y) \geq \min_{z \in I} \mathcal{R}(t, z), \forall t \in \mathcal{T}$$

$$\text{So } \mathcal{R}(t, i) \geq \min_{y \in I} \mathcal{R}(t, y), \forall i \in h(h(I)), \forall t \in \mathcal{T}.$$

$$\mu_{I\downarrow}(t) = \inf_{y \in \mathcal{I}} [\min(1, 1 + \mathcal{R}(t, y) - \mu_I(y))] = \min_{y \in I} \mathcal{R}(t, y)$$

$$\text{So } \mu_{I\uparrow\downarrow}(i) = \inf_{t \in \mathcal{T}} [\min(1, 1 + \mathcal{R}(t, i) - \mu_{I\downarrow}(t))], \forall i \in h(h(I))$$

$$\mu_{I\uparrow\downarrow}(i) = \inf_{t \in \mathcal{T}} [\min(1, 1 + \mathcal{R}(t, i) - \min_{y \in I} \mathcal{R}(t, y))]$$

$$\mu_{I\uparrow\downarrow}(i) = 1$$

Thus $i \in h(I)$ and $h(h(I)) \subset h(I)$.

So $h(I) = h(h(I))$.

Therefore, h is a closure operator.

□

Appendix D

Fuzzy Close Algorithm: Experimental Results

In this chapter, we present a benchmark that enables to assess *fuzzy Close*. In order to compare our algorithm to the fuzzy version of Apriori (Agrawal et al., 1993) and to UBFFPT (Lin et al., 2010a), we have implemented these algorithms. As our implementations may not be fully optimized, our results do not show any execution time. The metric that we used is the number of database passes. It allows to directly compare the fuzzy version of Apriori to our algorithm.

D.1 Datasets

We used three different datasets. The first one is the *mushroom* dataset (Schlimmer, 1987). It contains 8124 instances (or transactions). The number of features (or items) is 22. Those are all categorical features, so the final binary dataset contains 119 attributes. To fuzzify it, zeros were replaced by a uniform random number in $[0, 0.5[$ and ones were replaced by a uniform random number in $[0.5, 1]$.

The two other datasets come from the 2017 Civil Service People Survey (Government of the United Kingdom, 2017). Those are surveys that only contain numbers in $[0, 1]$. One dataset, that is called *benchmark scores*, contains 9 instances. Features have been pruned to avoid missing values for a final amount of 87 features. The other dataset is called *all organisation scores*. After filtering missing values, the dataset contains 93 instances and 84 features.

D.2 Results and Discussion

Results are shown in Figure D.1. For the mushroom dataset, we can observe that our algorithm makes at best one less database pass than the fuzzy version of Apriori. This is due to the fact that data are not highly correlated and are sparse. That means that most frequent itemsets are closed. As a consequence, with the cost of computing closures, our algorithm should not be expected to outperform Apriori and UBFFPT on such a dataset.

Observations are different with the two other datasets. We can see that the lower the minimum support threshold, the larger the difference between the number of database passes of both algorithms. These data come from surveys, whose data are usually highly correlated and dense. Our algorithm takes advantage of this using the closure operator. Thus, most generators are much shorter than their closures. That explains the lower amount of database passes.

The UBFFPT algorithm needs 4 database passes to construct its tree and to extract frequent itemsets. Besides, frequent pattern mining algorithms, such as UBFFPT,

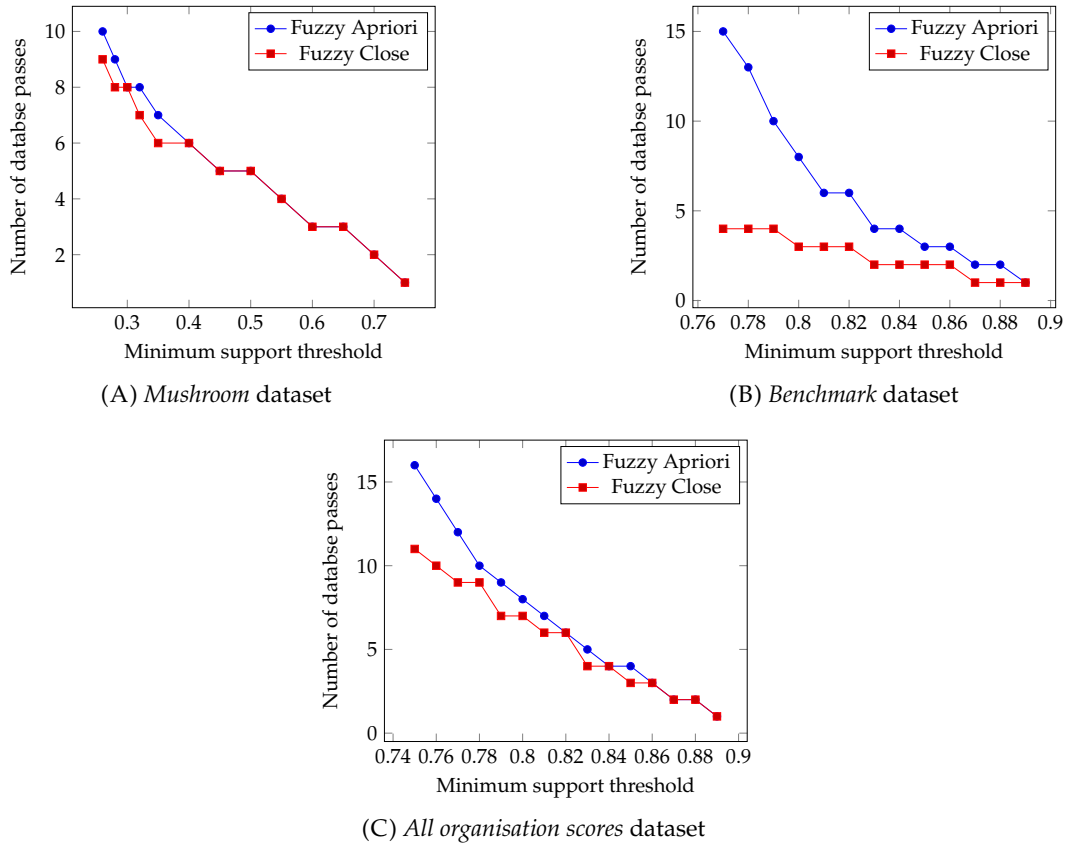


FIGURE D.1: Plots showing the number of database passes relatively to the minimum support threshold for the three datasets.

spend most of their time traversing the tree. For highly correlated data, as in the benchmark dataset, our algorithm has an edge on these algorithms. Moreover, it consumes less memory than Apriori, which generates many candidates at each iteration, and than UBFFPT, which browses all the paths to the item under study¹ to generate candidates.

Also, the first iteration of generating closures in our algorithm can bring valuable insight. Indeed, if most 1-itemsets are closed, then the data is likely to be weakly correlated and another algorithm is likely to perform better. However, if the proportion of closed 1-itemsets is low, the data is likely to be highly correlated and our algorithm will then compute all the frequent itemsets in few database passes.

¹One item is usually represented by several nodes in the tree.

Appendix E

Topological Sorting

In this chapter, we focus on defining what topological sorting is. We also give an illustrative example and then mention the main algorithms for obtaining a topological sort.

E.1 Definitions

Given a directed acyclic graph $G = (V, E)$, a *topological sort* on G is a total order of its vertices, which is defined as:

Definition 52: total order

A dyadic relation \leq_A is a *total order* on a set A if $\forall a_1, a_2, a_3 \in A$ it verifies:

- if $a_1 \leq_A a_2$ and $a_2 \leq_A a_1$, then $a_1 = a_2$ (antisymmetry),
- if $a_1 \leq_A a_2$ and $a_2 \leq_A a_3$, then $a_1 \leq_A a_3$ (transitivity),
- $a_1 \leq_A a_2$ or $a_2 \leq_A a_1$ (connexity).

We can thus define what a *topological sort* is.

Definition 53: topological sort

A *topological sort* of a directed acyclic graph (V, E) is a total order of its vertices such that for each edge $(v_1, v_2) \in E$, v_1 comes before v_2 in the ordering.

It should be noted that topological sorting requires a directed *acyclic* graph. If the graph under study contains a circle, then a total order on the vertices cannot be obtained.

E.2 Example

Let us consider the graph $G = (V, E)$ with $V = \{A, B, C, D, E, F\}$. It is displayed on figure E.1 (on the next page). This graph admits several topological sorts:

- $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F$
- $A \rightarrow B \rightarrow D \rightarrow C \rightarrow E \rightarrow F$
- $A \rightarrow B \rightarrow D \rightarrow C \rightarrow F \rightarrow E$
- $A \rightarrow B \rightarrow C \rightarrow D \rightarrow F \rightarrow E$
- $A \rightarrow B \rightarrow C \rightarrow E \rightarrow D \rightarrow F$

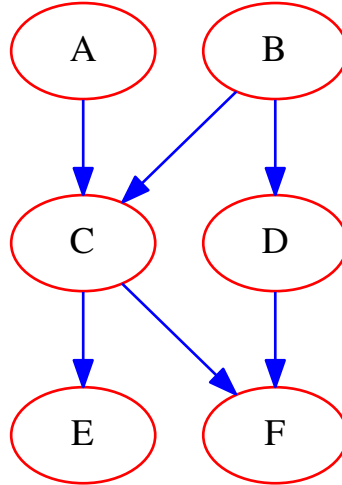


FIGURE E.1: Example of a directed acyclic graph for topological sorting.

- $B \rightarrow A \rightarrow C \rightarrow D \rightarrow E \rightarrow F$
- $B \rightarrow A \rightarrow D \rightarrow C \rightarrow E \rightarrow F$
- $B \rightarrow A \rightarrow D \rightarrow C \rightarrow F \rightarrow E$
- $B \rightarrow A \rightarrow C \rightarrow D \rightarrow F \rightarrow E$
- $B \rightarrow A \rightarrow C \rightarrow E \rightarrow D \rightarrow F$
- $B \rightarrow D \rightarrow A \rightarrow C \rightarrow E \rightarrow F$
- $B \rightarrow D \rightarrow A \rightarrow C \rightarrow F \rightarrow E$

E.3 Algorithms

There are two main algorithms for obtaining a topological sort. The first one, *Kahn's algorithm* (Kahn, 1962), starts by searching all the nodes that have no incoming edges. Then, it loops over these nodes, adds them to the tail of the sort and goes down edge by edge. When a new node is visited, the incoming edge that led to it is removed and, if it has no other incoming edge, then it is added to the tail of the sort. This algorithm has a linear complexity in the number of nodes and the number of edges.

The other algorithm relies on depth-first search. It has been first introduced by (Tarjan, 1976). The first node to visit is selected randomly. Then, starting from this node, it performs a depth-first search that finishes when a node has no outgoing edges or when the node has already been visited. Then, another unvisited node is selected and the same process is applied until all nodes have been visited.

Appendix F

Additional Results

F.1 Constraints in the Organ Annotation Experiment

In this section, we specify the constraints that enabled to generate the explanations in figure 9.15 on page 145.

$$\begin{aligned}
 C = \{ & (x_{l_lung}, x_{r_lung}, \mathcal{R}_{\text{completely to the left of}}), \\
 & (x_{l_kidney}, x_{spleen}, \mathcal{R}_{\text{below}}), \\
 & (x_{r_psoas}, x_{r_kidney}, \mathcal{R}_{\text{below}}), \\
 & (x_{l_kidney}, x_{r_kidney}, \mathcal{R}_{\text{to the left of}}), \\
 & (x_{l_kidney}, x_{l_lung}, \mathcal{R}_{\text{completely below}}), \\
 & (x_{r_kidney}, x_{r_lung}, \mathcal{R}_{\text{completely below}}), \\
 & (x_{l_lung}, x_{l_psoas}, \mathcal{R}_{\text{above}}), \\
 & (x_{spleen}, x_{r_lung}, \mathcal{R}_{\text{to the left of}}), \\
 & (x_{liver}, x_{spleen}, \mathcal{R}_{\text{completely to the right of}}), \\
 & (x_{l_psoas}, x_{r_psoas}, \mathcal{R}_{\text{completely to the left of}}), \\
 & (x_{l_lung}, x_{spleen}, \mathcal{R}_{\text{above}}), \\
 & (x_{r_kidney}, x_{spleen}, \mathcal{R}_{\text{to the right of}}), \\
 & (x_{r_psoas}, x_{r_lung}, \mathcal{R}_{\text{completely below}}), \\
 & (x_{spleen}, x_{r_kidney}, \mathcal{R}_{\text{to the left of}}), \\
 & (x_{spleen}, x_{l_lung}, \mathcal{R}_{\text{completely below}}), \\
 & (x_{r_lung}, x_{l_lung}, \mathcal{R}_{\text{completely to the right of}}), \\
 & (x_{r_kidney}, x_{liver}, \mathcal{R}_{\text{completely below}}), \\
 & (x_{r_lung}, x_{r_psoas}, \mathcal{R}_{\text{above}}), \\
 & (x_{l_psoas}, x_{l_lung}, \mathcal{R}_{\text{completely below}}), \\
 & (x_{r_psoas}, x_{liver}, \mathcal{R}_{\text{below}}), \\
 & (x_{bladder}, x_{r_lung}, \mathcal{R}_{\text{below}}), \\
 & (x_{liver}, x_{r_lung}, \mathcal{R}_{\text{below}}), \\
 & (x_{r_psoas}, x_{l_psoas}, \mathcal{R}_{\text{completely to the right of}}), \\
 & (x_{r_kidney}, x_{l_psoas}, \mathcal{R}_{\text{to the right of}}), \\
 & (x_{spleen}, x_{liver}, \mathcal{R}_{\text{completely to the left of}}), \\
 & (x_{l_psoas}, x_{spleen}, \mathcal{R}_{\text{below}}), \\
 & (x_{l_kidney}, x_{l_psoas}, \mathcal{R}_{\text{above}}) \}
 \end{aligned}$$

F.2 Time Series Classification: Linguistic Variables

In this section, we specify the linguistic variables used in the experiment about time series classification of toxic chemicals (cf. Chapter 10).

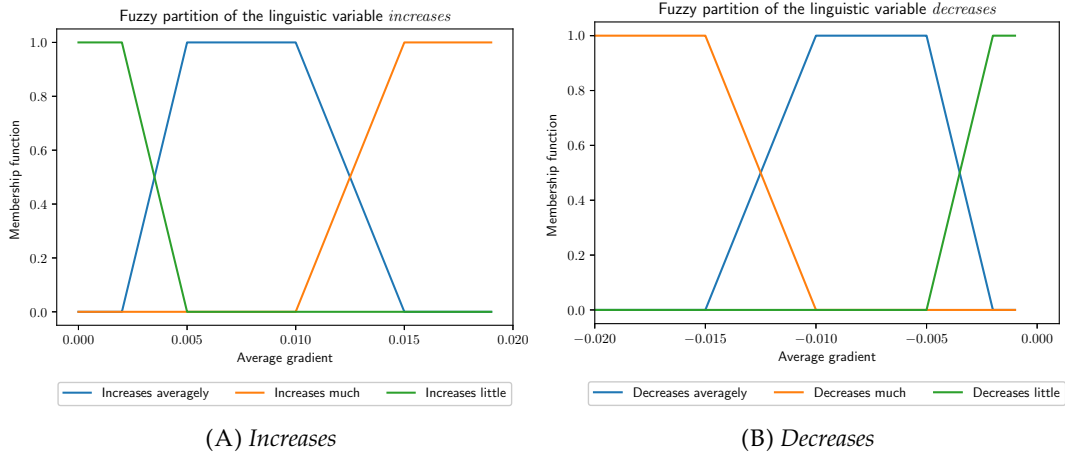


FIGURE F.1: Fuzzy sets corresponding to the linguistic variables *Increases* and *Decreases*.

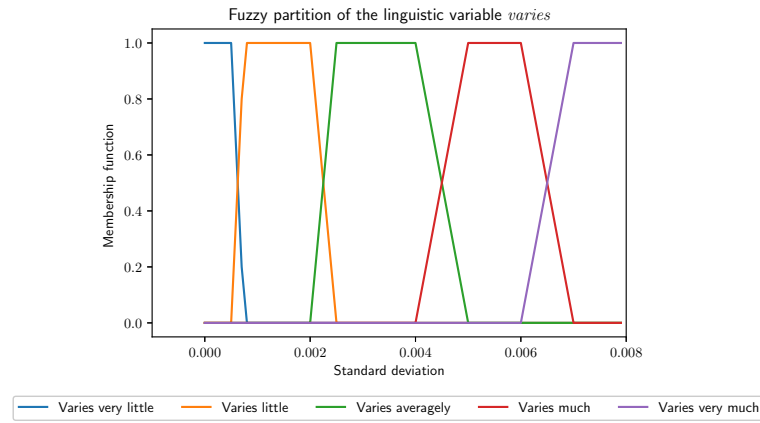


FIGURE F.2: Fuzzy sets corresponding to the linguistic variable *Varies*.

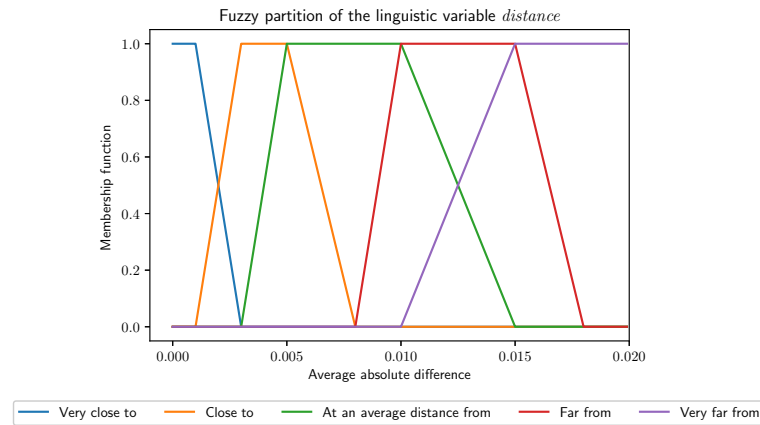


FIGURE F.3: Fuzzy sets corresponding to the linguistic variable *Distance*.

Appendix G

SIMD

Single Instruction Multiple Data (SIMD) is a vectorization paradigm. It enables to implement instructions that perform parallel computations in a single clock cycle. Figure G.1 presents an example that shows how parallelization through SIMD instructions can help to perform more operations than non-SIMD instructions. Such instructions are available in most microprocessors from Intel and AMD as a set of vector extensions named *Advanced Vector Extensions* (AVX).

In this appendix, we briefly explain the architecture on which AVX relies and we then give examples of instructions.

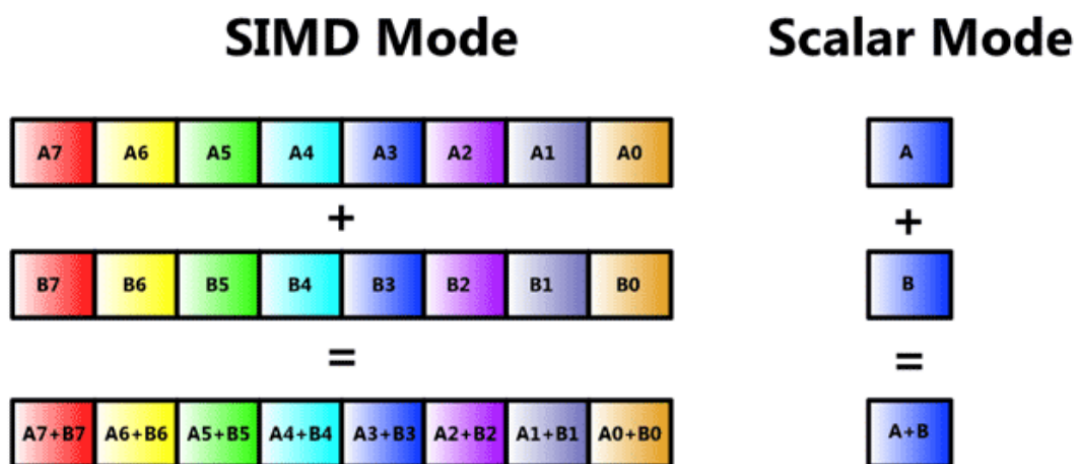


FIGURE G.1: Figure representing how SIMD instructions enable to parallelize computations compared to a single instruction operating on single data (Intel, 2011).

G.1 Architecture

SIMD instructions rely on a register on which operations are performed. For AVX, the width of the register is 128 bits. Two other versions have been released in the last few years: AVX2, which is based on a 256-bit register, and AVX512, whose register is twice as big and reaches 512 bits. The experiments presented in Chapter 8 were performed using these three variants of AVX.

In Figure G.2, we see how instances from different data types fit in the register. In Chapter 8, we dealt with 8-bit unsigned integer and used AVX512. Thus, we were able to process 64 instances per instruction.

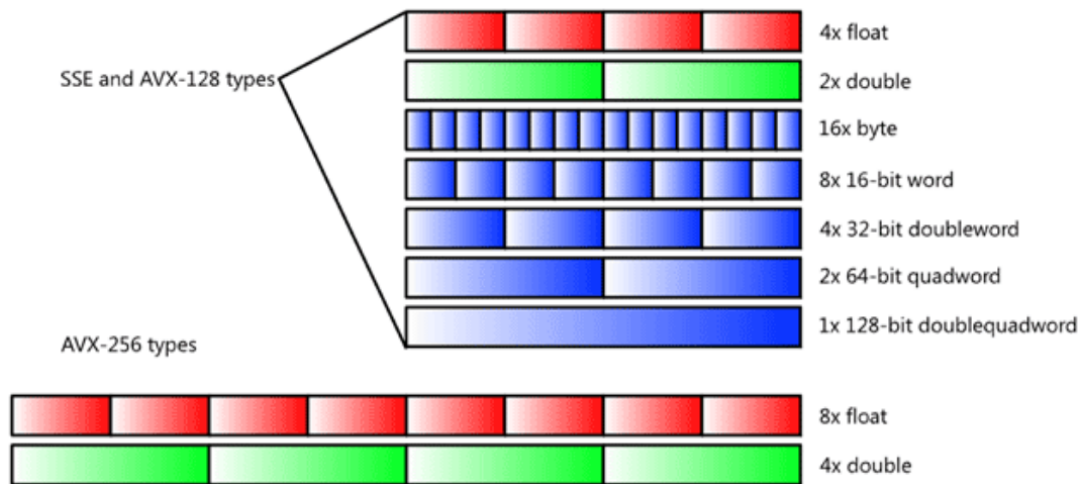


FIGURE G.2: Figure showing how different data types fit in a 128-bit register (upper part of the figure) and in a 256-bit register (lower part) (Intel, 2011).

G.2 Instructions

We resorted to three different instructions in our experiments:

- `_mm512_loadu_si512`, which enables to load 512 bits of integer data from memory into the register,
- `_mm512_storeu_si512`, which enables to store 512 bits of integer data into the register,
- `_mm512_min_epu8`, which enables to return the minimum value between two 8-bit unsigned integers,
- `_mm512_max_epu8`, which enables to return the maximum value between two 8-bit unsigned integers.

Appendix H

Résumé en français

Ces dernières années, en raison d'importantes améliorations des performances, l'Intelligence Artificielle (IA) s'est démocratisée au sein de notre société. En effet, de nombreuses applications s'appuyant sur des IA sont désormais disponibles et ont un impact concret sur des milliards d'êtres humains. Cependant, les modèles utilisés aujourd'hui sont opaques, ce qui empêche de véritablement comprendre le raisonnement qu'ils effectuent pour produire un résultat. Cette faiblesse est d'autant plus importante qu'il a été montré que ces modèles se basent parfois sur des corrélations fallacieuses présentes dans les données d'entraînement et peuvent donc produire des raisonnements erronés. Ainsi, dans certains domaines sensibles pour lesquels le coût d'une mauvaise décision est très élevé, tels que la médecine ou les voitures autonomes par exemple, il est dangereux de s'appuyer sur une IA dont on ne peut pas déterminer précisément le comportement.

Dans cette thèse, dans le cadre des applications sensibles, nous proposons une approche permettant de construire un modèle de classification ou d'annotation capable de fournir une explication claire et détaillée du raisonnement qui l'a mené à un résultat. En particulier, le modèle s'appuie sur la détection de relations entre les entités présentes dans les instances de la base de données étudiée. Ainsi, le modèle peut produire un résultat et générer une explication en langage naturel qui fait intervenir ces relations.

Étant donné un vocabulaire de relations floues potentiellement pertinentes fourni par un expert, l'approche que nous proposons se divise en trois grandes étapes : i) les relations du vocabulaire sont évaluées sur les données d'entraînement de manière à éviter les calculs redondants et inutiles ; ii) les sous-ensembles de relations les plus fréquents au sein de la base d'entraînement sont extraits ; iii) des règles (dans le cadre de la classification) ou des contraintes pour définir un problème de satisfaction de contraintes floues (dans le cadre de l'annotation) sont générées à partir des sous-ensembles de relations extraits à l'étape précédente afin de construire un modèle. Ce modèle peut ensuite être utilisé pour classer ou annoter de nouvelles instances et pour générer des explications en langage naturel retranscrivant son raisonnement et la manière avec laquelle il utilise les relations apprises.

Dans ces travaux, nous proposons notamment un nouvel algorithme de fouille de données pour extraire les motifs fréquents au sein d'une base de données floues. Lorsque les données sont fortement corrélées entre elles, ce qui est le cas au sein d'une même classe d'entités par exemple, cet algorithme tire profit d'un opérateur de fermeture pour être plus performant. Nous présentons également deux nouvelles heuristiques pour évaluer rapidement les relations de notre vocabulaire sur les données d'entraînement. La première heuristique proposée consiste à éliminer, pendant le processus d'entraînement, les relations qui ne pourront plus être considérées comme fréquentes par notre algorithme de fouille de données. La seconde

heuristique s'intéresse aux liens logiques et aux dépendances entre les relations utilisées pour obtenir un ordre d'évaluation sur ces relations afin d'éviter les calculs redondants. Nous présentons également un nouvel algorithme de calcul de la dilatation floue. Cet algorithme tient compte uniquement des pixels contribuant au résultat final et s'appuie sur des instructions SIMD afin de paralléliser le calcul au maximum.

Nous avons testé notre approche sur une base de données jouet pour la classification et sur un jeu d'images médicales pour faire de l'annotation d'organes. Outre les performances en classification/annotation, nous avons validé la pertinence des explications produites en demandant à un panel de participants d'évaluer plusieurs critères, tels que la cohérence des explications, leurs lisibilités ou encore la confiance qu'elles procurent. Nous avons également évalué les performances de l'algorithme de dilatation floue que nous proposons. Contrairement à l'état de l'art, il produit un résultat exact tout en étant plus rapide sur les tailles d'objet présentes dans nos jeux de données.

Bibliography

- Aamodt, A. and Plaza, E. (1994). "Case-based reasoning: Foundational issues, methodological variations, and system approaches". In: *AI communications* 7.1, pp. 39–59.
- Adebayo, J. et al. (2018). "Sanity Checks for Saliency Maps". In: *Advances in Neural Information Processing Systems* 31, pp. 9505–9515.
- Agrawal, R., Imieliński, T., and Swami, A. (1993). "Mining association rules between sets of items in large databases". In: *Acm sigmod record*. Vol. 22. 2, pp. 207–216.
- Allen, J. F. (1983). "Maintaining knowledge about temporal intervals". In: *Communications of the ACM* 26.11, pp. 832–843.
- Alonso, J.M. and Bugarín, A. (2019). "ExpliClas: Automatic Generation of Explanations in Natural Language for Weka Classifiers". In: *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1–6.
- Alvarez Melis, D. and Jaakkola, T. (2018). "Towards Robust Interpretability with Self-Explaining Neural Networks". In: *Advances in Neural Information Processing Systems* 31, pp. 7775–7784.
- Assemblée Nationale (2019). *Projet de loi relatif à la bioéthique*. URL: <http://www.assemblee-nationale.fr/15/projets/pl2187-ei.asp>.
- Au, W-H. and Chan, K. C. C. (1998). "An effective algorithm for discovering fuzzy rules in relational databases". In: *Fuzzy Systems Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*. Vol. 2, pp. 1314–1319.
- Baader, F. (1996). "A Formal Definition for the Expressive Power of Terminological Knowledge Representation Languages". In: *Journal of Logic and Computation* 6.1, pp. 33–54.
- Baaj, I. and Poli, J-P. (2019). "Natural Language Generation of Explanations of Fuzzy Inference Decisions". In: *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*.
- Baehrens, D. et al. (2010). "How to explain individual classification decisions". In: *Journal of Machine Learning Research* 11.Jun, pp. 1803–1831.
- Belohlávek, R. (1999). "Fuzzy Galois Connections". In: *Mathematical Logic Quarterly* 45.4, pp. 497–504.
- (2001). "Fuzzy closure operators". In: *Journal of mathematical analysis and applications* 262.2, pp. 473–489.
- (2012). *Fuzzy relational systems: foundations and principles*. Vol. 20.
- Biederman, I. (1981). *On the Semantics of a Glance at a Scene*.
- Biran, O. and Cotton, C. (2017). "Explanation and justification in machine learning: A survey". In: *IJCAI 2017 Workshop on Explainable Artificial Intelligence*.
- Biran, O. and McKeown, K. (2017). "Human-Centric Justification of Machine Learning Predictions". In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp. 1461–1467.
- Bloch, I. (1999a). "Fuzzy relative position between objects in image processing: a morphological approach". In: *IEEE transactions on pattern analysis and machine intelligence* 21.7, pp. 657–664.

- Bloch, I. (1999b). "On fuzzy distances and their use in image processing under imprecision". In: *Pattern Recognition* 32.11, pp. 1873–1895.
- (2005). "Fuzzy spatial relationships for image processing and interpretation: a review". In: *Image and Vision Computing* 23.2, pp. 89–110.
- Bloch, I. and Maitre, H. (1995). "Fuzzy mathematical morphologies: A comparative study". In: *Pattern Recognition* 28.9, pp. 1341–1387.
- Booth, S., Muise, C., and Shah, J. (2019). "Evaluating the Interpretability of the Knowledge Compilation Map: Communicating Logical Statements Effectively". In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 5801–5807.
- Borgefors, G. (1986). "Distance transformations in digital images". In: *Computer Vision, Graphics, and Image Processing* 34.3, pp. 344–371.
- Borgida, A. (1996). "On the relative expressiveness of description logics and predicate logics". In: *Artificial Intelligence* 82.1, pp. 353–367.
- Breiman, L. (1984). *Classification and regression trees*.
- (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32.
- Brinker, T. J. et al. (2019). "Deep learning outperformed 136 of 157 dermatologists in a head-to-head dermoscopic melanoma image classification task". In: *European Journal of Cancer* 113, pp. 47–54.
- Bruce, G. (1983). "Principles of rule-based expert systems". In: *Advances in computers* 22, pp. 163–216.
- Buciluă, C., Caruana, R., and Niculescu-Mizil, A. (2006). "Model compression". In: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541.
- Byrne, R. M. J. (2019). "Counterfactuals in Explainable Artificial Intelligence (XAI): Evidence from Human Reasoning". In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 6276–6282.
- Byrne, R. M. J. and Johnson-Laird, P. N. (2009). "'If' and the problems of conditional reasoning". In: *Trends in Cognitive Sciences* 13.7, pp. 282–287.
- Cariñena, P. et al. (2000). "A language for expressing fuzzy temporal rules." In: *Mathware and Soft Computing* 7.2-3, pp. 213–227.
- Cawley, G. C. and Talbot, N. L. C. (2010). "On over-fitting in model selection and subsequent selection bias in performance evaluation". In: *Journal of Machine Learning Research* 11.Jul, pp. 2079–2107.
- Cayrol, M., Farreny, H., and Prade, H. (1982). "Fuzzy pattern matching". In: *Kybernetes* 11.2, pp. 103–116.
- Chanussot, J., Nyström, I., and Sladoje, N. (2005). "Shape signatures of fuzzy star-shaped sets based on distance from the centroid". In: *Pattern Recognition Letters* 26.6, pp. 735–746.
- Chen, J. et al. (2018a). "Knowledge-based transfer learning explanation". In: *Sixteenth International Conference on Principles of Knowledge Representation and Reasoning*.
- Chen, J. et al. (2018b). "Learning to Explain: An Information-Theoretic Perspective on Model Interpretation". In: *Proceedings of the 35th International Conference on Machine Learning*. Vol. 80, pp. 883–892.
- Chen, T. and Guestrin, C. (2016). "Xgboost: A scalable tree boosting system". In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794.
- Cinbis, R. G. and Aksoy, S. (2007). "Relative Position-Based Spatial Relationships using Mathematical Morphology". In: *2007 IEEE International Conference on Image Processing*. Vol. 2, pp. 97–100.

- Clément, M., Kurtz, C., and Wendling, L. (2018). "Learning spatial relations and shapes for structural object description and scene recognition". In: *Pattern Recognition* 84, pp. 197–210.
- Clément, M. et al. (2017). "Directional Enlacement Histograms for the Description of Complex Spatial Configurations between Objects". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.12, pp. 2366–2380.
- Cohen, N., Sharir, O., and Shashua, A. (2016). "On the Expressive Power of Deep Learning: A Tensor Analysis". In: *29th Annual Conference on Learning Theory*. Vol. 49, pp. 698–728.
- Colliot, O. (2003). "Représentation, évaluation et utilisation de relations spatiales pour l'interprétation d'images. Application à la reconnaissance de structures anatomiques en imagerie médicale". PhD thesis. Télécom ParisTech.
- Craven, M. and Shavlik, J. W. (1996). "Extracting tree-structured representations of trained networks". In: *Advances in neural information processing systems*, pp. 24–30.
- Dagum, L. and Menon, R. (1998). "OpenMP: an industry standard API for shared-memory programming". In: *Computational Science & Engineering, IEEE* 5.1, pp. 46–55.
- Dai, B., Zhang, Y., and Lin, D. (2017). "Detecting visual relationships with deep relational networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3076–3086.
- Das, A. et al. (2017). "Human attention in visual question answering: Do humans and deep networks look at the same regions?" In: *Computer Vision and Image Understanding* 163, pp. 90–100.
- Davis, R., Shrobe, H., and Szolovits, P. (1993). "What is a knowledge representation?" In: *AI magazine* 14.1, pp. 17–17.
- De Raedt, L. and Thon, I. (2011). "Probabilistic Rule Learning". In: *Inductive Logic Programming*, pp. 47–58.
- Donadello, I., Serafini, L., and Garcez, A. d'Avila (2017). "Logic Tensor Networks for Semantic Image Interpretation". In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp. 1596–1602.
- Doshi-Velez, F. and Kim, B. (2017). "Towards A Rigorous Science of Interpretable Machine Learning". In: *eprint arXiv:1702.08608*.
- Dubois, D., Fargier, H., and Prade, H. (1996). "Possibility theory in constraint satisfaction problems: Handling priority, preference and uncertainty". In: *Applied Intelligence* 6.4, pp. 287–309.
- Dubois, D., HadjAli, A., and Prade, H. (2003). "Fuzziness and uncertainty in temporal reasoning". In: *J. UCS* 9.9, p. 1168.
- Dubois, D., Prade, H., and Testemale, C. (1988). "Weighted fuzzy pattern matching". In: *Fuzzy sets and systems* 28.3, pp. 313–331.
- Dumancic, S., Garcia-Duran, A., and Niepert, M. (2019a). "A Comparative Study of Distributional and Symbolic Paradigms for Relational Learning". In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 6088–6094.
- (2019b). "A Comparative Study of Distributional and Symbolic Paradigms for Relational Learning". In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 6088–6094.
- Dumancic, S. et al. (2019c). "Learning Relational Representations with Auto-encoding Logic Programs". In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 6081–6087.
- Dwyer, K. and Holte, R. (2007). "Decision tree instability and active learning". In: *European Conference on Machine Learning*, pp. 128–139.

- Erhan, D. et al. (2009). "Visualizing higher-layer features of a deep network". In: European Council (2016). *The general data protection regulation*.
- Evans, R. and Grefenstette, E. (2018). "Learning explanatory rules from noisy data". In: *Journal of Artificial Intelligence Research* 61, pp. 1–64.
- Feigenbaum, E. A., Buchanan, B. G., and Lederberg, J. (1970). "On generality and problem solving: A case study using the DENDRAL program". In:
- Fernandez, A. et al. (2019). "Evolutionary Fuzzy Systems for Explainable Artificial Intelligence: Why, When, What for, and Where to?" In: *IEEE Computational Intelligence Magazine* 14.1, pp. 69–81.
- FICO (1989). FICO. URL: <https://www.fico.com>.
- (2018). FICO Explainable Machine Learning Challenge. URL: <https://community.fico.com/s/explainable-machine-learning-challenge>.
- Forgy, C. L. (1989). "Rete: A fast algorithm for the many pattern/many object pattern match problem". In: *Readings in Artificial Intelligence and Databases*, pp. 547–559.
- Fournier-Viger, P. et al. (2017). "A survey of itemset mining". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 7.4, e1207.
- Francis, J., Rahbarnia, F., and Matsakis, P. (2018). "Fuzzy NLG system for extensive verbal description of relative positions". In: *2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1404–1411.
- Freeman, J. (1975). "The modelling of spatial relations". In: *Computer Graphics and Image Processing* 4.2, pp. 156–171.
- Friedman, J. H. (2001). "Greedy function approximation: a gradient boosting machine". In: *Annals of statistics*, pp. 1189–1232.
- Frosst, N. and Hinton, G. (2017). "Distilling a neural network into a soft decision tree". In: *arXiv preprint arXiv:1711.09784*.
- Ganter, B. and Wille, R. (1996). *Formal concept analysis: mathematical foundations*.
- Gatt, A. and Reiter, E. (2009). "SimpleNLG: A Realisation Engine for Practical Applications". In: *Proceedings of the 12th European Workshop on Natural Language Generation*, pp. 90–93.
- Gerla, G. (2013). *Fuzzy logic: mathematical tools for approximate reasoning*. Vol. 11.
- Geurts, P. (2001). "Pattern Extraction for Time Series Classification". In: *Principles of Data Mining and Knowledge Discovery*, pp. 115–127.
- Ghorbani, A. et al. (2019). "Towards automatic concept-based explanations". In: *Advances in Neural Information Processing Systems*, pp. 9273–9282.
- Gilpin, L. H. et al. (2018). "Explaining Explanations: An Overview of Interpretability of Machine Learning". In: *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 80–89.
- Ginet, C. (2016). "Reasons Explanation: Further Defense of a Non-causal Account". In: *The Journal of Ethics* 20.1-3, pp. 219–228.
- Goldstein, A. et al. (2015). "Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation". In: *Journal of Computational and Graphical Statistics* 24.1, pp. 44–65.
- Gondra, I. and Cabria, I. (2016). "Computing force field-based directional maps in subquadratic time". In: *Knowledge-Based Systems* 95, pp. 58–70.
- González, A. et al. (2012). "An efficient inductive genetic learning algorithm for fuzzy relational rules". In: *International Journal of Computational Intelligence Systems* 5.2, pp. 212–230.
- Goodman, B. and Flaxman, S. (2017). "European Union regulations on algorithmic decision-making and a "right to explanation"". In: *AI Magazine* 38.3, pp. 50–57.
- Government of the United Kingdom (2017). URL: <https://www.gov.uk/government/publications/civil-service-people-survey-2017-results--2>.

- Goyal, Y. et al. (2019). "Counterfactual Visual Explanations". In: *arXiv preprint arXiv:1904.07451*.
- Graziani, M., Andrearczyk, V., and Müller, H. (2018). "Regression concept vectors for bidirectional explanations in histopathology". In: *Understanding and Interpreting Machine Learning in Medical Image Computing Applications*, pp. 124–132.
- Grosan, C. and Abraham, A. (2011). "Rule-based expert systems". In: *Intelligent Systems*, pp. 149–185.
- Guidotti, R. et al. (2018). "Local rule-based explanations of black box decision systems". In: *arXiv preprint arXiv:1805.10820*.
- Gunning, D. (2016). "Explainable artificial intelligence (xAI)". In:
- Haenssle, H. A. et al. (2018). "Man against machine: diagnostic performance of a deep learning convolutional neural network for dermoscopic melanoma recognition in comparison to 58 dermatologists". In: *Annals of Oncology* 29.8, pp. 1836–1842.
- Hallac, D., Nystrup, P., and Boyd, S. (2019). "Greedy Gaussian segmentation of multivariate time series". In: *Advances in Data Analysis and Classification* 13.3, pp. 727–751.
- Halpern, J. Y. and Pearl, J. (2001). "Causes and Explanations: A Structural-Model Approach". In: *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*.
- Han, J., Pei, J., and Yin, Y. (2000). "Mining frequent patterns without candidate generation". In: *ACM sigmod record* 29.2, pp. 1–12.
- Hastie, T. and Tibshirani, R. (1986). "Generalized Additive Models". In: *Statistical Science*, pp. 297–310.
- Hendricks, L. A. et al. (2016). "Generating visual explanations". In: *European Conference on Computer Vision*, pp. 3–19.
- Hendricks, L. A. et al. (2018). "Grounding visual explanations". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 264–279.
- Hernández, D., Clementini, E., and Di Felice, P. (1995). "Qualitative distances". In: *International Conference on Spatial Information Theory*, pp. 45–57.
- Hinton, G., Vinyals, O., and Dean, J. (2015). "Distilling the knowledge in a neural network". In: *arXiv preprint arXiv:1503.02531*.
- Ho, T. K. (1995). "Random decision forests". In: *Proceedings of 3rd international conference on document analysis and recognition*. Vol. 1, pp. 278–282.
- Hochreiter, S. and Schmidhuber, J. (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780.
- Hoffman, Robert R et al. (2018). "Metrics for explainable AI: Challenges and prospects". In: *arXiv preprint arXiv:1812.04608*.
- Höhle, U. (1996). "On the fundamentals of fuzzy set theory". In: *Journal of mathematical Analysis and Applications* 201.3, pp. 786–826.
- Hong, T-P, Kuo, C-S., and Chi, S-C. (1999). "Mining association rules from quantitative data". In: *Intelligent data analysis* 3.5, pp. 363–376.
- Hotel, O. (2017). "Algorithms, methods and models for the application of surface acoustic wave sensors to the recognition of chemical compound signatures". PhD thesis. Université Pierre et Marie Curie - Paris VI.
- Hudelot, C., Atif, J., and Bloch, I. (2008). "Fuzzy spatial relation ontology for image interpretation". In: *Fuzzy Sets and Systems* 159.15, pp. 1929–1951.
- Huk Park, D. et al. (2018). "Multimodal explanations: Justifying decisions and pointing to the evidence". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8779–8788.

- Ignatiev, A., Narodytska, N., and Marques-Silva, J. (2019). "Abduction-based explanations for machine learning models". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33, pp. 1511–1519.
- Intel (2011). *Introduction to Intel® Advanced Vector Extensions*. <https://software.intel.com/en-us/articles/introduction-to-intel-advanced-vector-extensions>.
- Irsoy, O., Yıldız, O. T., and Alpaydın, E. (2012). "Soft decision trees". In: *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pp. 1819–1822.
- Jakulin, A. et al. (2005). "Nomograms for visualizing support vector machines". In: *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pp. 108–117.
- James, G. et al. (2013). *An introduction to statistical learning*. Vol. 112.
- Jimenez-del-Toro, O. et al. (2016). "Cloud-based evaluation of anatomical structure segmentation and landmark detection algorithms: VISCERAL anatomy benchmarks". In: *IEEE transactions on medical imaging* 35.11, pp. 2459–2475.
- Jin, Y. (2000). "Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement". In: *IEEE Transactions on Fuzzy Systems* 8.2, pp. 212–221.
- Johnson-Laird, P. N. (1983). *Mental models: Towards a cognitive science of language, inference, and consciousness*. 6.
- (2010). "Mental models and human reasoning". In: *Proceedings of the National Academy of Sciences* 107.43, pp. 18243–18250.
- Kahn, A. B. (1962). "Topological Sorting of Large Networks". In: *Commun. ACM* 5.11, pp. 558–562.
- Kellogg, R. T. (1980). "Feature frequency and hypothesis testing in the acquisition of rule-governed concepts". In: *Memory & Cognition* 8.3, pp. 297–303.
- Kenny, E. M. and Keane, M. T. (2019). "Twin-Systems to Explain Artificial Neural Networks using Case-Based Reasoning: Comparative Tests of Feature-Weighting Methods in ANN-CBR Twins for XAI". In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 2708–2715.
- Kim, B., Khanna, R., and Koyejo, O. O. (2016). "Examples are not enough, learn to criticize! Criticism for Interpretability". In: *Advances in Neural Information Processing Systems* 29, pp. 2280–2288.
- Kim, B., Rudin, C., and Shah, J. (2014). "The Bayesian Case Model: A Generative Approach for Case-Based Reasoning and Prototype Classification". In: *Advances in Neural Information Processing Systems* 27, pp. 1952–1960.
- Kim, B. et al. (2018). "Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV)". In: *International Conference on Machine Learning*, pp. 2673–2682.
- Kindermans, P-J. et al. (2017). "Learning how to explain neural networks: Patternnet and patternattribution". In: *arXiv preprint arXiv:1705.05598*.
- Kindermans, P-J. et al. (2019). "The (un) reliability of saliency methods". In: *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pp. 267–280.
- Koh, P. W. and Liang, P. (2017). "Understanding black-box predictions via influence functions". In: *Proceedings of the 34th International Conference on Machine Learning—Volume 70*, pp. 1885–1894.
- Kramer, S., Pfahringer, B., and Helma, C. (1998). "Stochastic propositionalization of non-determinate background knowledge". In: *International Conference on Inductive Logic Programming*, pp. 80–94.
- Kuok, C. M., Fu, A., and Wong, M. H. (1998). "Mining fuzzy association rules in databases". In: *ACM Sigmod Record* 27.1, pp. 41–46.

- Lacassagne, L. et al. (2014). "High Level Transforms for SIMD and low-level computer vision algorithms". In: *ACM Workshop on Programming Models for SIMD/Vector Processing (PPoPP)*, pp. 49–56.
- An Evaluation of the Human-Interpretability of Explanation* (2018).
- Lakkaraju, H., Bach, S. H., and Leskovec, J. (2016). "Interpretable decision sets: A joint framework for description and prediction". In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1675–1684.
- Lakkaraju, H. et al. (2019). "Faithful and customizable explanations of black box models". In: *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 131–138.
- Langs, G. et al. (2013). "VISCERAL: Towards Large Data in Medical Imaging - Challenges and Directions". In: *MCBR-CDS MICCAI workshop*. Vol. 7723.
- Laugel, T. et al. (2019). "The Dangers of Post-hoc Interpretability: Unjustified Counterfactual Explanations". In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 2801–2807.
- Le Yaouanc, J-M. and Poli, J-P. (2012). "A Fuzzy Spatio-temporal-Based Approach for Activity Recognition". In: *Advances in Conceptual Modeling*, pp. 314–323.
- Lécué, F. and Pommellet, T. (2019). "Feeding Machine Learning with Knowledge Graphs for Explainable Object Detection". In: *Proceedings of the ISWC 2019 Satellite Tracks (Posters & Demonstrations, Industry, and Outrageous Ideas) co-located with 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26-30, 2019*, pp. 277–280.
- Lesot, M-J., Rifqi, M., and Bouchon-Meunier, B. (2008). "Fuzzy prototypes: From a cognitive view to a machine learning principle". In: *Fuzzy Sets and Their Extensions: Representation, Aggregation and Models*, pp. 431–452.
- Letham, B. et al. (2015). "Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model". In: *The Annals of Applied Statistics* 9.3, pp. 1350–1371.
- Levesque, H. J. and Brachman, R. J. (1987). "Expressiveness and tractability in knowledge representation and reasoning 1". In: *Computational intelligence* 3.1, pp. 78–93.
- Li, O. et al. (2018). "Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions". In: *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Likert, R. (1932). "A technique for the measurement of attitudes." In: *Archives of psychology*.
- Lin, C-W., Hong, T-P., and Lu, W-H. (2010a). "A two-phase fuzzy mining approach". In: *Fuzzy Systems (FUZZ), 2010 IEEE International Conference on*, pp. 1–5.
- (2010b). "An efficient tree-based fuzzy data mining approach". In: *International Journal of Fuzzy Systems* 12.2, pp. 150–157.
- Lindsay, R. K. et al. (1993). "DENDRAL: A case study of the first expert system for scientific hypothesis formation". In: *Artificial Intelligence* 61.2, pp. 209–261.
- Lipton, Z. C. (2018). "The Mythos of Model Interpretability". In: *Queue* 16.3, 30:57.
- Lloyd, J. R. and Ghahramani, Z. (2015). "Statistical model criticism using kernel two sample tests". In: *Advances in Neural Information Processing Systems*, pp. 829–837.
- Lu, C. et al. (2016). "Visual relationship detection with language priors". In: *European Conference on Computer Vision*, pp. 852–869.
- Lundberg, S. M. and Lee, S-I. (2017). "A Unified Approach to Interpreting Model Predictions". In: *Advances in Neural Information Processing Systems* 30, pp. 4765–4774.

- Lundberg, S. M. et al. (2019). "Explainable AI for Trees: From Local Explanations to Global Understanding". In: *arXiv preprint arXiv:1905.04610*.
- Mackworth, A. K. (1977). "Consistency in networks of relations". In: *Artificial Intelligence* 8.1, pp. 99–118.
- Malioutov, D. M. et al. (2017). "Learning Interpretable Classification Rules with Boolean Compressed Sensing". In: *Transparent Data Mining for Big and Small Data*, pp. 95–121.
- Marcus, G. (2018). "Deep Learning: A Critical Appraisal". In: *CoRR* abs/1801.00631.
- Matsakis, P. (2002). "Understanding the Spatial Organization of Image Regions by Means of Force Histograms: A Guided Tour". In: *Applying Soft Computing in Defining Spatial Relations*, pp. 99–122.
- Matsakis, P., Naeem, M., and Rahbarnia, F. (2015). "Introducing the Φ -Descriptor - A Most Versatile Relative Position Descriptor". In: *ICPRAM*.
- Matsakis, P., Ni, J., and Veltman, M. (2009). "Directional relationships to a reference object: A quantitative approach based on force fields". In: *2009 16th IEEE International Conference on Image Processing (ICIP)*, pp. 321–324.
- Matsakis, P., Ni, J., and Wang, X. (2006). "Object Localization Based on Directional Information: Case of 2D Raster Data". In: *18th International Conference on Pattern Recognition (ICPR'06)*. Vol. 2, pp. 142–146.
- Matsakis, P. and Wendling, L. (1999). "A new way to represent the relative position between areal objects". In: *IEEE Transactions on pattern analysis and machine intelligence* 21.7, pp. 634–643.
- McClure, J. (2002). "Goal-based Explanations of Actions and Outcomes". In: *European Review of Social Psychology* 12.1, pp. 201–235.
- Michalski, R. S. (1983). "A theory and methodology of inductive learning". In: *Machine learning*, pp. 83–134.
- Miller, T. (2017). "Explanation in Artificial Intelligence: Insights from the Social Sciences". In: *CoRR* abs/1706.07269.
- Miyajima, K. and Ralescu, A. (1994a). "Spatial organization in 2D images". In: *Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the Third IEEE Conference on*, pp. 100–105.
- (1994b). "Spatial organization in 2D segmented images: representation and recognition of primitive spatial relations". In: *Fuzzy Sets and Systems* 65.2-3, pp. 225–236.
- Molnar, C. (2019). *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable*. <https://christophm.github.io/interpretable-ml-book/>.
- Montanari, U. (1974). "Networks of Constraints: Fundamental Properties and Applications to Picture Processing". In: 7, pp. 95–132.
- Montavon, G. et al. (2017). "Explaining nonlinear classification decisions with deep taylor decomposition". In: *Pattern Recognition* 65, pp. 211–222.
- Moosavi-Dezfooli, S-H., Fawzi, A., and Frossard, P. (2016). "Deepfool: a simple and accurate method to fool deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582.
- Nelder, J. A. and Mead, R. (1965). "A simplex method for function minimization". In: *The computer journal* 7.4, pp. 308–313.
- Nelder, J. A. and Wedderburn, R. W. M. (1972). "Generalized linear models". In: *Journal of the Royal Statistical Society: Series A (General)* 135.3, pp. 370–384.
- Nguyen, A., Yosinski, J., and Clune, J. (2015). "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 427–436.

- Olah, C., Mordvintsev, A., and Schubert, L. (2017). "Feature Visualization". In: *Distill*. <https://distill.pub/2017/feature-visualization>.
- Olah, C. et al. (2018). "The Building Blocks of Interpretability". In: *Distill*. <https://distill.pub/2018/building-blocks>.
- OpenCCG (2004). <https://github.com/OpenCCG/openccg>.
- Papadimitriou, S. and Mavroudi, S. (2005). "The fuzzy frequent pattern tree". In: *The WSEAS International Conference on Computers*, pp. 1–7.
- Papenmeier, A., Englebienne, G., and Seifert, C. (2019). "How model accuracy and explanation fidelity influence user trust in AI". In: *IJCAI Workshop on Explainable Artificial Intelligence (XAI) 2019*.
- Pasquier, N. et al. (1999). "Efficient mining of association rules using closed itemset lattices". In: *Information systems* 24.1, pp. 25–46.
- Paulheim, H. (2017). "Knowledge graph refinement: A survey of approaches and evaluation methods". In: *Semantic web* 8.3, pp. 489–508.
- Pedreschi, D. et al. (2019). "Meaningful explanations of black box AI decision systems". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33, pp. 9780–9784.
- Poli, J-P., Boudet, L., and Le Yaouanc, J-M. (2018). "Online spatio-temporal fuzzy relations". In: *2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 1003–1010.
- Poli, J-P., Boudet, L., and Mercier, D. (2016). "Online temporal reasoning for event and data streams processing". In: *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pp. 2257–2264.
- Poli, J-P. et al. (2017). "Online Fuzzy Temporal Operators for Complex System Monitoring". In: *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pp. 375–384.
- Poursabzi-Sangdeh, F. et al. (2018). "Manipulating and measuring model interpretability". In: *arXiv preprint arXiv:1802.07810*.
- Quinlan, J. R. (1986). "Induction of decision trees". In: *Machine learning* 1.1, pp. 81–106.
- (1993). *C4. 5: programs for machine learning*.
- Raghu, M. et al. (2017). "On the expressive power of deep neural networks". In: *Proceedings of the 34th International Conference on Machine Learning*.
- Randell, D. A., Cui, Z., and Cohn, A. G. (1992). "A spatial logic based on regions and connection." In: *Proceedings of the 1992 AAAI Conference on Artificial Intelligence*.
- Read, S. J. and Marcus-Newhall, A. (1993). "Explanatory coherence in social explanations: A parallel distributed processing account". In: *Journal of Personality and Social Psychology*.
- Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "Why should i trust you?: Explaining the predictions of any classifier". In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144.
- (2018). "Anchors: High-precision model-agnostic explanations". In: *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Rosenfeld, A. and Klette, R. (1985). "Degree of adjacency or surroundedness". In: *Pattern Recognition* 18.2, pp. 169–177.
- Rudin, C. (2019). "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead". In: *Nature Machine Intelligence* 1.5, pp. 206–215.
- Russakovsky, O. et al. (2015). "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision* 115.3, pp. 211–252.
- Schlegel, U. et al. (2019). "Towards a rigorous evaluation of XAI Methods on Time Series". In: *arXiv preprint arXiv:1909.07082*.

- Schlimmer, J. (1987). "Concept acquisition through representational adjustment". PhD thesis. Department of Information and Computer Science, University of California.
- Schockaert, S. and De Cock, M. (2008). "Temporal reasoning about fuzzy intervals". In: *Artificial Intelligence* 172.8, pp. 1158–1193.
- Schockaert, S., De Cock, M., and Kerre, E. E. (2008a). "Fuzzifying Allen's temporal interval relations". In: *IEEE Transactions on Fuzzy Systems* 16.2, pp. 517–533.
- (2009). "Spatial reasoning in a fuzzy region connection calculus". In: *Artificial Intelligence* 173.2, pp. 258–298.
- (2011). *Reasoning about fuzzy temporal and spatial information from the web*. Vol. 3.
- Schockaert, S. et al. (2008b). "Fuzzy region connection calculus: Representing vague topological information". In: *International Journal of Approximate Reasoning* 48.1, pp. 314–331.
- Selbst, A. D. and Powles, J. (2017). "Meaningful information and the right to explanation". In: *International Data Privacy Law* 7.4, pp. 233–242.
- Shapley, L. S. (1953). "A value for n-person games". In: *Contributions to the Theory of Games* 2.28, pp. 307–317.
- Shortliffe, E. H. and Buchanan, B. G. (1975). "A model of inexact reasoning in medicine". In: *Mathematical Biosciences* 23.3, pp. 351–379.
- Shrikumar, A., Greenside, P., and Kundaje, A. (2017). "Learning important features through propagating activation differences". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3145–3153.
- Silver, D. et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587, p. 484.
- Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps". In: *CoRR abs/1312.6034*.
- Slack, D. et al. (2019). *How can we fool LIME and SHAP? Adversarial Attacks on Post hoc Explanation Methods*. arXiv: [1911.02508](https://arxiv.org/abs/1911.02508).
- Springenberg, J. T. et al. (2014). "Striving for simplicity: The all convolutional net". In: *arXiv preprint arXiv:1412.6806*.
- Štrumbelj, E. and Kononenko, I. (2010). "An Efficient Explanation of Individual Classifications using Game Theory". In: *Journal of Machine Learning Research* 11, pp. 1–18.
- Sundararajan, M., Taly, A., and Yan, Q. (2017). "Axiomatic attribution for deep networks". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3319–3328.
- Szegedy, C. et al. (2013). "Intriguing properties of neural networks". In: *arXiv preprint arXiv:1312.6199*.
- Tarjan, R. E. (1976). "Edge-disjoint spanning trees and depth-first search". In: *Acta Informatica* 6.2, pp. 171–185.
- Tarski, A., Mostowski, A., and Robinson, R. M. (1953). *Undecidable theories*. Vol. 13.
- Thagard, P. (1989). "Explanatory coherence". In: *Behavioral and Brain Sciences* 12.3, pp. 435–467.
- Tibshirani, R. (1996). "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1, pp. 267–288.
- Turney, P. (1995). "Bias and the quantification of stability". In: *Machine Learning* 20.1–2, pp. 23–33.
- United States House of Representatives (1974). *Equal Credit Opportunity Act*.
- Uno, T., Kiyomi, M., and Arimura, H. (2004). "LCM ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets". In:

- Uspensky, J. V. (1937). *Introduction to Mathematical Probability*.
- Vanegas, M. C. (2011). "Spatial relations and spatial reasoning for the interpretation of Earth observation images using a structural model". PhD thesis.
- Vanegas, M. C., Bloch, I., and Inglada, J. (2012). "Alignment and parallelism for the description of high-resolution remote sensing images". In: *IEEE Transactions on Geoscience and Remote Sensing* 51.6, pp. 3542–3557.
- (2016). "Fuzzy constraint satisfaction problem for model-based image interpretation". In: *Fuzzy Sets and Systems* 286, pp. 1–29.
- Vasilyeva, N., Wilkenfeld, D. A., and Lombrozo, T. (2015). "Goals Affect the Perceived Quality of Explanations". In:
- Vaswani, A. et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems*, pp. 5998–6008.
- Wachter, S., Mittelstadt, B., and Floridi, L. (2017a). "Why a right to explanation of automated decision-making does not exist in the general data protection regulation". In: *International Data Privacy Law* 7.2, pp. 76–99.
- Wachter, S., Mittelstadt, B., and Russell, C. (2017b). "Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR". In: *Harv. JL & Tech.* 31, p. 841.
- Waltz, D. L. (1972). "Generating semantic descriptions from drawings of scenes with shadows". In:
- Wang, X., Ni, J., and Matsakis, P. (2006). "Fuzzy Object Localization Based on Directional (and Distance) Information". In: *2006 IEEE International Conference on Fuzzy Systems*, pp. 256–263.
- Wu, M. et al. (2018). "Beyond sparsity: Tree regularization of deep models for interpretability". In: *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Wu, X. et al. (2008). "Top 10 algorithms in data mining". In: *Knowledge and information systems* 14.1, pp. 1–37.
- Yager, R. R. (1991). "The representation of fuzzy relational production rules". In: *Applied Intelligence* 1.1, pp. 35–42.
- Yager, R. R. and Filev, D. P. (1996). "Relational partitioning of fuzzy rules". In: *Fuzzy sets and systems* 80.1, pp. 57–69.
- Yosinski, J. et al. (2015). "Understanding Neural Networks through deep Visualization". In: *arXiv preprint arXiv:1506.06579*.
- Zadeh, L. A. (1965). "Fuzzy sets". In: *Information and Control* 8.3, pp. 338–353.
- (1975). "The concept of a linguistic variable and its application to approximate reasoning—I". In: *Information sciences* 8.3, pp. 199–249.
- (1982). "A note on prototype theory and fuzzy sets". In: *Cognition* 12.3, p. 291.
- (1996). "Fuzzy sets and information granularity". In: *Fuzzy Sets, Fuzzy Logic, And Fuzzy Systems: Selected Papers by Lotfi A Zadeh*, pp. 433–448.
- (1999). "Fuzzy logic= computing with words". In: *Computing with Words in Information/Intelligent Systems* 1, pp. 3–23.
- Zaki, M. J. (2000). "Scalable algorithms for association mining". In: *IEEE Transactions on Knowledge and Data Engineering* 12.3, pp. 372–390.
- Zeiler, M. D. and Fergus, R. (2014). "Visualizing and understanding convolutional networks". In: *European conference on computer vision*, pp. 818–833.
- Zhang, Q. et al. (2019). "Interpreting CNNs via decision trees". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6261–6270.
- Zhang, X. et al. (2018). "Interpretable Deep Learning under Fire". In: *arXiv preprint arXiv:1812.00891*.
- Zhou, B. et al. (2018). "Interpretable basis decomposition for visual explanation". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 119–134.

Zucker, J.-D. and Ganascia, J.-G. (1996). "Representation Changes for Efficient Learning in Structural Domains". In: *ICML*.

Titre : Classification et annotation explicable par apprentissage de relations et raisonnement

Mots clés : Intelligence artificielle explicable, Apprentissage de relations, Logique floue

Résumé : Avec les succès récents de l'apprentissage profond et les interactions toujours plus nombreuses entre êtres humains et intelligences artificielles, l'explicabilité est devenue une préoccupation majeure. En effet, il est difficile de comprendre le comportement des réseaux de neurones profonds, ce qui les rend inadaptés à une utilisation dans les systèmes critiques. Dans cette thèse, nous proposons une approche visant à classer ou annoter des signaux tout en expliquant les résultats obtenus. Elle est basée sur l'utilisation d'un modèle transparent, dont le raisonnement est clair, et de relations floues interprétables qui permettent de représenter l'imprécision du langage naturel. Au lieu d'apprendre sur des exemples sur lesquels les relations ont été annotées, nous proposons

de définir un ensemble de relations au préalable. L'évaluation de ces relations sur les exemples de la base d'entraînement est accélérée grâce à deux heuristiques que nous présentons. Ensuite, les relations les plus pertinentes sont extraites en utilisant un nouvel algorithme de frequent itemset mining flou. Ces relations permettent de construire des règles pour la classification ou des contraintes pour l'annotation. Ainsi, une explication en langage naturel peut être générée. Nous présentons des expériences sur des images et des séries temporelles afin de montrer la généralité de notre approche. En particulier, son application à l'annotation d'organe explicable a été bien évaluée par un ensemble de participants qui ont jugé les explications convaincantes et cohérentes.

Title: Explainable classification and annotation through relation learning and reasoning

Keywords: Explainable artificial intelligence, Relational learning, Fuzzy logic

Abstract: With the recent successes of deep learning and the growing interactions between humans and AIs, explainability issues have risen. Indeed, it is difficult to understand the behaviour of deep neural networks and thus such opaque models are not suited for high-stake applications. In this thesis, we propose an approach for performing classification or annotation and providing explanations. It is based on a transparent model, whose reasoning is clear, and on interpretable fuzzy relations that enable to express the vagueness of natural language. Instead of learning on training instances that are annotated with relations, we propose to rely on a set of relations that was set beforehand.

We present two heuristics that make the process of evaluating relations faster. Then, the most relevant relations can be extracted using a new fuzzy frequent itemset mining algorithm. These relations enable to build rules, for classification, and constraints, for annotation. Since the strengths of our approach are the transparency of the model and the interpretability of the relations, an explanation in natural language can be generated. We present experiments on images and time series that show the genericity of the approach. In particular, the application to explainable organ annotation was received positively by a set of participants that judges the explanations consistent and convincing.