Touch interface for guided authoring of expert systems rules

Jean-Philippe Poli, Jean-Paul Laurent

Touch Interface For Guided Authoring Of Expert Systems Rules

Jean-Philippe Poli, Jean-Paul Laurent
CEA, LIST, Data Analysis and System Intelligence Laboratory,
91191 Gif-sur-Yvette cedex, France.
Email: firstname.lastname@cea.fr

Abstract

In recent years, artificial intelligence tools have democratized and are more and more often used by people who are not experts in the field.

For instance, systems based on rules or constraints require human expertise as input to replicate the desired behavior. Despite the explosion of new devices and new input paradigms, such as tablets and other touch interfaces, usability of these tools seems not to have taken advantage of these recent advances.

In this article, we focus on a fuzzy expert system for which users want to enter rules. We use our industrial partnerships to define with current users their needs in terms of rule authoring. They expressed their will of more mobility, more modernism, less mathematics. We present our work that involves the use of new touch interfaces to capture a fuzzy rule with only one finger. We end this article by the evaluation of the GUI with a user panel.

1 Introduction

At the birth of artificial intelligence (AI), AI specialists and specialists of Human-Computer Interaction (HCI) opposed in their approach: the first worked on machines which were intended to be equal to Humans and the latter wanted to increase Humans capabilities [1]. Nowadays, AI tools are democratizing and are intended to assist humans in their daily work. They affect very different audiences, which are often not specialists in these algorithms. While AI techniques have developed and improved over the years, it is only recently that researchers interested in what has been called "the usability of AI" [2]. Indeed, if a technology is to be used in a real context by the general public, effective user interaction is a major aspect of the acceptance of this technology [3]. For instance, Dadzie and Petrelli [4] indicate that despite powerful recommendation algorithms, commercial websites such as *Amazon* had to test several methods for displaying results before reaching users acceptance.

In this article, we present our work on a rule-based system. In such systems, it is necessary to collect the knowledge so that inference engines can reproduce the reasoning of a human expert. Knowledge authoring is often a tedious step and as shown in previous work section, input interfaces have only changed a little. This explains our motivation to propose a rule composition graphical user interface (GUI) on modern touch devices like tablets and which uses a widespread mode of interaction, the *drag-and-drop*. Indeed, *drag-and-drop* on touch devices is an interaction technique whose result is directly visible, allowing to apply less attention and fewer cognitive abilities [5]. Moreover, recent studies show that *drag-and-drop* is an intuitive gesture that allows great precision interaction for users with various profiles and various ages, thanks to the continuous contact with the surface [6]. This mode of interaction thus seems interesting to study.

Without loss of generality, we illustrate our work on rules authoring in a fuzzy expert system in the following section. We then interest in previous work described in the literature. The following sections are devoted to the description of our GUI and its assessment by a panel of users. Finally, we conclude with an open conclusion about the prospects of this work.

2 Relative work

In this section, we are interested in the work of GUI that were previously conducted and which are applicable to the authoring of rules in an expert system. In fuzzy logic, both linguistic variables and rules have to be defined. We consider the inputs and outputs of the system are known and defined in advance in order to focus on authoring rules.

The methods from the literature can be classified into two main categories depending on whether they consist of a textual or graphical representation; speech, to our knowledge, has never been applied to that.

On the one hand, text based methods consist in natural language processing techniques [7] and directly extract the rules from a text. From a user perspective, these methods have the advantage of proposing to write rules directly in his language, with its own vocabulary. Algorithms analyze the text and extract causal relationships to obtain the rules. These methods have been applied to the extraction of fuzzy rules for the nursing community [8] and to online texts [9]. The understanding of free text is still a research topic and rules extraction techniques are quite limited, so it is quite possible that the user enters a text devoid of semantics. To overcome this difficulty, it is possible to constrain the user by exploiting *autocomplete* techniques [10]. In an even more compelled style, specific languages can be introduced, like intermediary languages whose syntax is more or less constrained, as the work of Acampora and Loia on a representation language for fuzzy rules based on XML [11]. However, the dexterity of the users on virtual keyboards is lower compared to physical keyboards [12].

On the other hand, graphical methods offer representations with less text, which symbolize the logical links between the components of the rules. This is

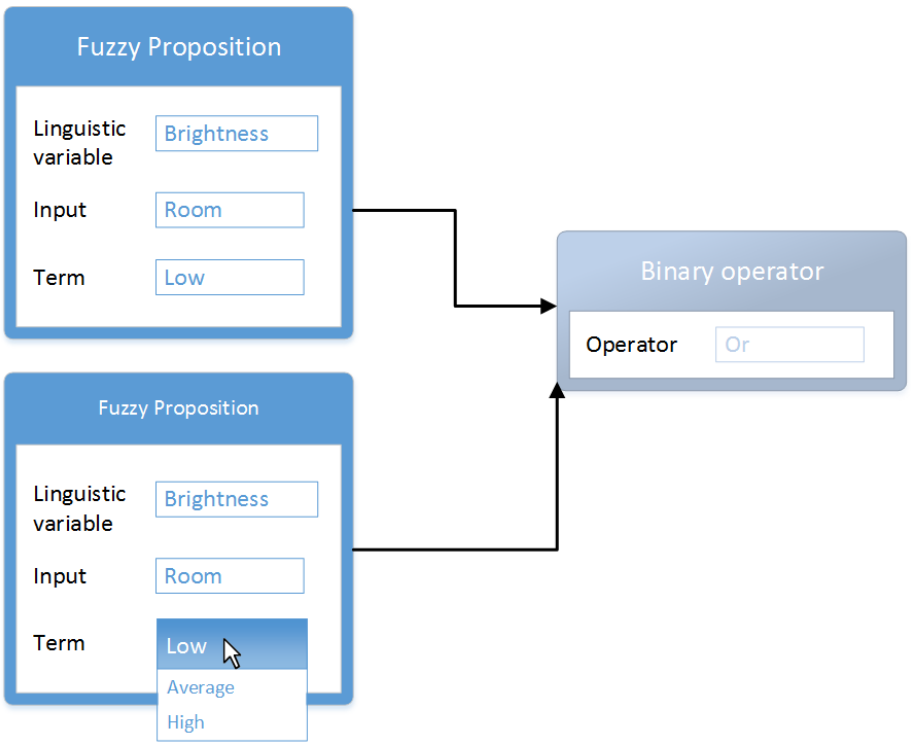

Figure 1: Example of the *Rule flowchart* tool.

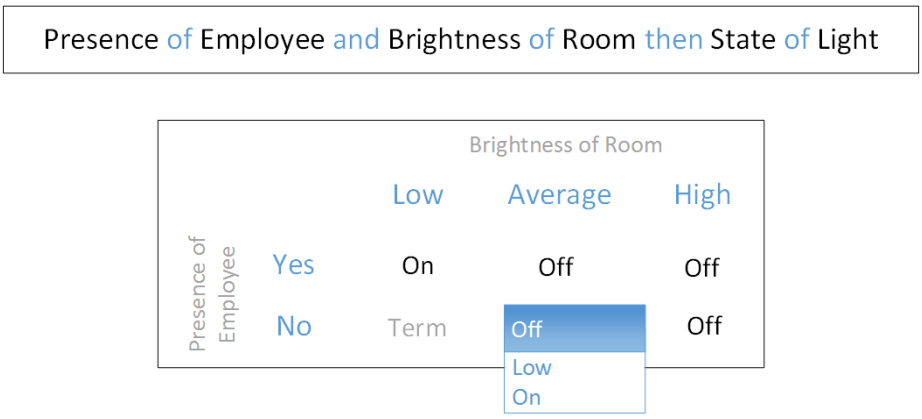

Figure 2: Example of the *Rule matrix* tool with 6 rules.

the case of flowcharts which represent each part of a rule by different boxes, potentially distinguishable by their shapes and colors. The user just has to connect each of these boxes to mark the logical link between them. Figure 1 shows an example of flowchart used for writing a rule: the rule consists of two propositions, each connected to an input and a binary operator, etc. The rule is displayed in the form "`IF` $proposition_1$ `OPERATOR` $proposition_2$ `THEN` *conclusion*". Rule authoring techniques based on flowcharts are both research topics, like in the work of Mosconi and Porta [13], and tools that can be found in commercial rule-based systems (in classical logic) such as IBM *ILOG* and BOSCH *Visual Rule Modeler*. Moreover, flowcharts can be easily adopted on touch devices, and as the reading is easy and natural, they can handle rules which are more complex to express textually [14]. However, users are not all equal using the flowcharts, which require a certain level of abstraction. This level of abstraction is easily found among graduate users but less among undergraduates (see section Evaluation).

The association matrices are another way to author rules, but in the form of a two-dimensional array. These interfaces take two inputs and provide a visually simple association between the occurrence of the values of the two inputs and the conclusion of the rule. Figure 2 shows an example of an association matrix: columns and lines represent the terms of two linguistic variables *Presence* and *Brightness*, whose terms are respectively Yes, No and Low, $Average$, $High$, and cells contain the chosen term of the linguistic variable $StateofLight$ whose terms are On, Low, Off. *Presence* and *Brightness* are input variables and $StateofLight$ is the output variable. This simple matrix referred to 8 rules:

- `IF` *Presence* `is` Yes `AND` *Brightness* `is` Low `THEN` $StateofLight$ `is` On;
- `IF` *Presence* `is` Yes `AND` *Brightness* `is` *Average* `THEN` $StateofLight$ `is` Off;
- etc...

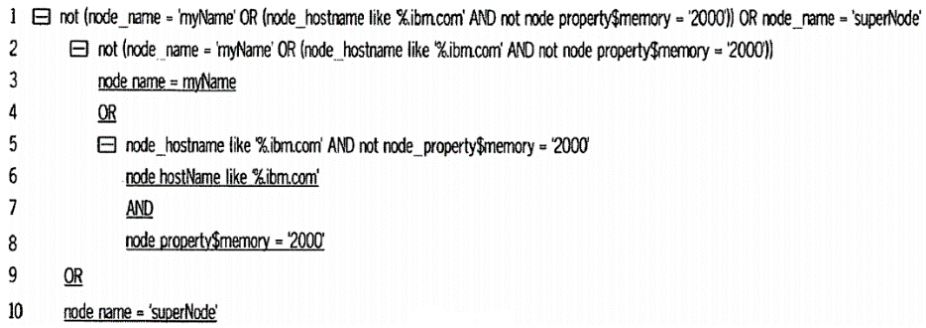

Figure 3: Treeview of a rule base [15].

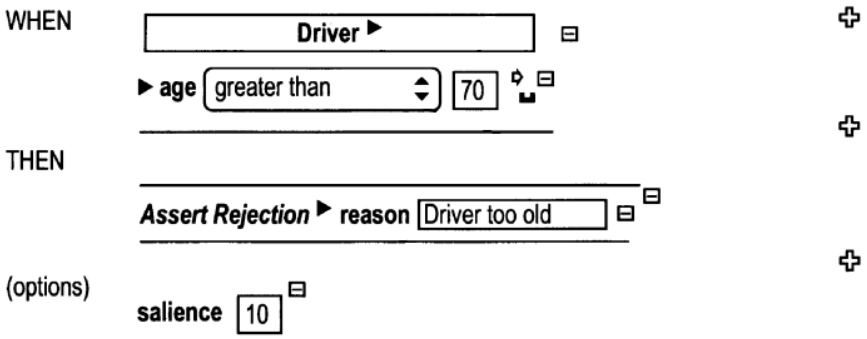

Figure 4: Rule patterns and empty fields [16].

Thus, the association matrices allow to enter a group of related rules very quickly. Some commercial software like *fuzzyTECH* use them as rule authoring software interfaces. The advantage is to force the completeness of the rulebase, since an empty cell in the association matrix represents a case that will not be processed by the inference engine. However, matrices are limited by their visualization in two dimensions which means that they can only handle two input variables.

Beyond these conventional graphical tools, other interfaces have been introduced to improve the user experience. We found tools based on *drag-and-drop* in a patent of IBM [15] in which the rule base is represented as a tree in which the user can expand or hide the branches (Figure 3) and can move items from one branch to another using *drag-and-drop*. Despite this, the rules no longer read as text which can disrupt users. Another patent of Red Hat [16] suggests the use of patterns of rules in which the user can select the available values (Figure 4). In this case, rule authoring reduces to choose a textual pattern and to fill editable fields, which allows an easy reading of the rule. These patterns are also the limits of the interface because only certain predefined rules can

be drawn. The same concept is found in natural language programming, for example, applied to home automation. In [17,18], interfaces are used to control the actuators installed in a home with an intuitive interface to build rules and usage scenarios. Note that the terms used in the rules are strongly linked to the application and we thus find the same limitations as the work of [16]. However, as the interface is dedicated to a given application, it is an effective way to author rules.

The following section outlines our motivations as well as the functionality of the rule editor that we implemented.

3 Tactile rule editor

3.1 Motivations

In the previous section, we introduced the conventional methods to author rules and more sophisticated methods that are transposable to the particular case of fuzzy logic. Those interfaces are generally a compromise between the interpretability of the rules by the system and the freedom given to the user during the authoring of the rules. A dilemma remains unresolved: moving away from a textual representation makes the users confused whereas text capture is not practical on touch devices.

In the two patents cited later, two ideas emerge: the use of *drag-and-drop*, with which users seem comfortable, and the selection of items predefined contextually.

Our approach was to meet our industrial partners, users of the fuzzy expert system we have developed, and imagine together what can be a convenient GUI. Globally, they want a more modern GUI, with a focus on mobility, less mathematical or logic. Regarding the state of the art, our contribution is about new devices with capacitive touchscreens. As previously stated, *drag-and-drop* on such screens are used to target an audience of different profiles and ages. This is also why we want to keep a form close to natural language while offering functionality similar to the interfaces described above.

Another difficulty comes from the nature of expert systems. Expert systems consist in a generic inference engine, but domain-specific rules. Thus, they can be applied to different domains and the rules can be authored by different kinds of users. Contrary to specific interfaces (as in [17,18]), the rule editor has to be as generic as possible.

3.2 Main features

In this article, we consider capturing rules assuming that the inputs and outputs of the system have already been created. For greater flexibility, we separated the concept of linguistic variable into an input and a measured physical quantity. This will eventually allow to factor definitions and to have multiple sources for the same measured magnitude.

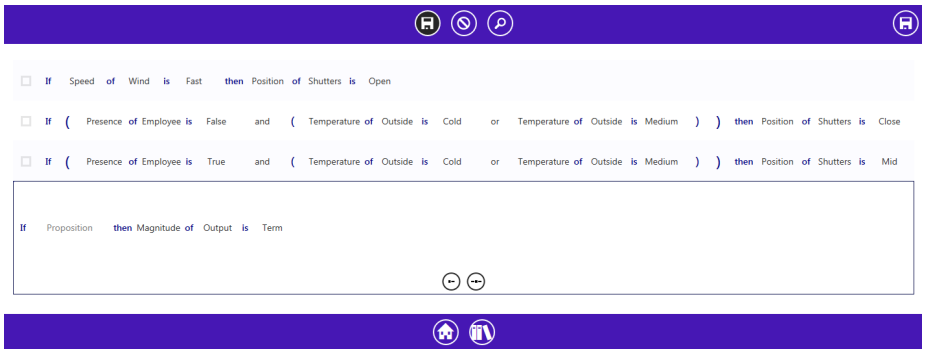

Figure 5: Overview of the *Rule editor* tool.

Our GUI offers to author a rule with a "filling in the blanks" design so that all users can easily and quickly produce rules. The blanks form a pattern that we call a "ghost", in reference to the pale color it takes as long as it is not completed. It is possible to choose a general form of rule, then the types of propositions that compose the rule. It is also possible to select and move the different parts of a rule, or insert a new operator. The insertion of a new operator may lead to the insertion of a ghost that describe the form that should take the missing part of the current expression. All these operations are done by *drag-and-drop*. Figure 5 presents a view of the GUI with a rule being edited, in which appears the ghost of a proposition, until a proposition pattern is chosen.

At this stage, the GUI would be too permissive allowing to move operators or expressions anywhere. We have adopted various measures to ensure that the user is guided while composing rules: *feedforwards* show possible actions and *feedbacks* help to focus on the result of the last action.

3.2.1 Inputs and outputs definition

Once the domain has been chosen, the user can select the shape of the curve to be inserted (figure 6). The curve is inserted in the middle of the domain. When the curve is selected, handles appear: the user has just to move the handles to change the parameters of the curve (figure 7). In the case of a multiline curve, a click can add a point and break the selected line. Some users asked us to add a way to precisely parametrize a curve. A click on a handle makes appear a box to type the value of the parameters or the coordinate of the point for a trapezoid, triangular or multiline membership function.

3.2.2 Adding a rule

To create a rule, the user has just to click on the button "Add" materialized by a "+" symbol. The rule being edited is highlighted in a box while leaving other rules visible. The rule is then materialized by a textual pattern containing a ghost of the premise proposition and of the conclusion. Clicking the premise

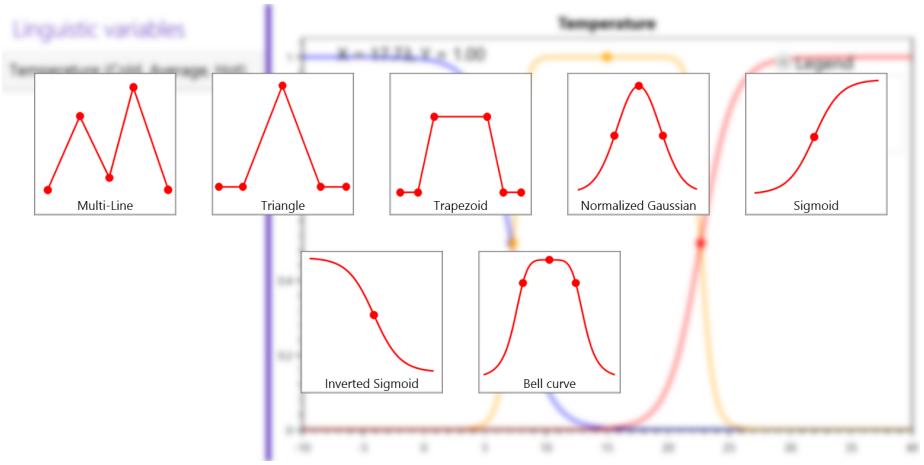

Figure 6: The choice of shapes for a membership function.

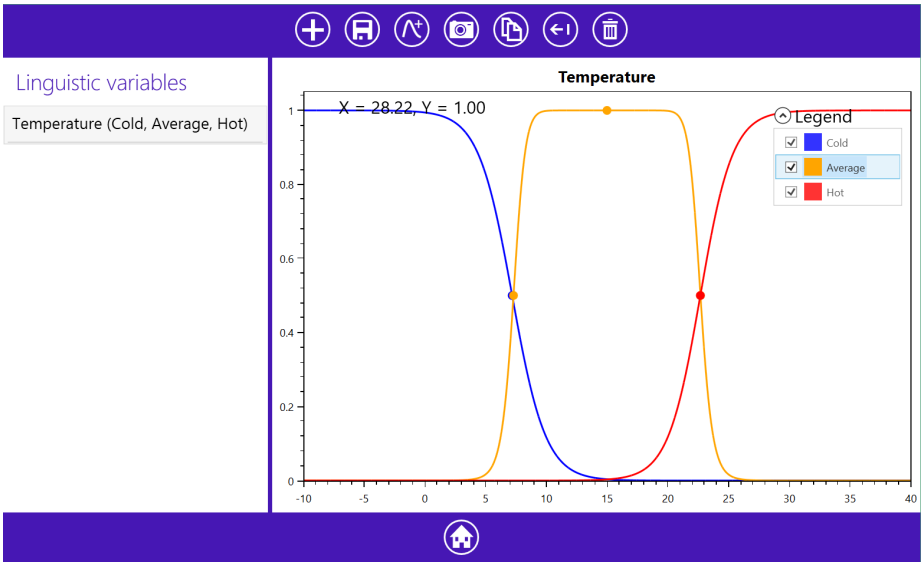

Figure 7: Example of membership function. The points are the handles to move.

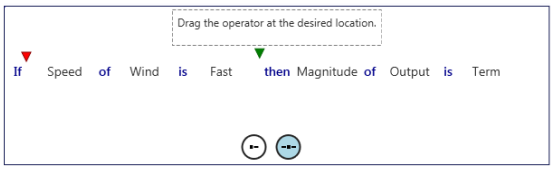

Figure 8: Focus on the possible destinations while *drag-and-drop* of the binary operator.

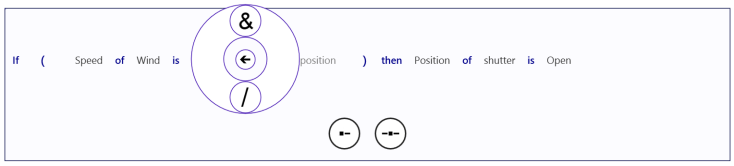

Figure 9: Pie menu for binary operator selection.

ghost makes appear a pie menu to select the desired type of proposition, either in the form `magnitude OF input IS adjective` or the form `input EQUALS value`. Each customizable term (in lower case in the examples above) can be changed with the help of a list of options. The editor verifies that the selected magnitude can be measured from the selected input and thus displayed only relevant choices in the list. Figure 5 shows a rule pattern with its various ghosts "Proposition", "Magnitude", "Output" and "Term".

3.2.3 Composition of expressions

You can create an expression (unary or binary) by dragging its icon to a portion of the premise of the rule. Once the icon is dropped in a correct place, a ghost of operator is placed, eventually followed by a ghost of an expression if the operator needs more operands. Animated red arrows indicate where the operator may be dropped during the drag phase (Figure 8).

Clicking the ghost "Op." of the operator makes appear a pie menu which shows the available operators (Figure 9). After the selection, the ghost is replaced with the selected operator. To change the operator, the user can simply click on it again to call back the pie menu, in which all operators are represented by icons. If a mouse is detected and if it hovers upon the icon, it displays a context-sensitive help message. If there is no mouse, a long-press on the icon makes appear the tooltip.

3.2.4 Moving parts of rules

We opted for a classical selection system based on a selection rectangle. To do this, the editor checks that the selected portion is consistent before permitting its displacement, that is to say the selection represents a well-formed expression.

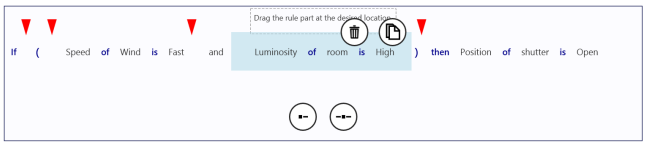

Figure 10: Available actions after the selection of a sub-expression.

Then contextual buttons appear to copy the portion of expression or delete it. It is also possible to *drag-and-drop* the selection to another place in the rule. Possible locations are indicated by animated red arrows (Figure 10).

In the following section, we describe the evaluation of this interface.

4 Evaluation

We first describe the experimental protocol and the panel of users. Then we discuss the results.

4.1 Protocol

In the literature, certain methods of rule entry have already been compared: we can mention for example the comparison of flowcharts and association matrices [19] or the comparison of a structured language based on English language and flowcharts and matrices [20]. In our case, we naturally confront our tool to other existing tools. Our choice fell on flowcharts and association matrices since they are the most used and the most accessible tools. We have thus developed two tools (*Rule matrix* and *Rule flowchart*) in addition to the one described in this article (*Rule editor*). We paid attention to their appearance so that they are as close as possible. The comparison between the three tools was made using the following criteria: time to adopt the tools, average time for rule authoring and number of erroneous rules. These quantitative measures are supplemented by qualitative information from a questionnaire given to the participants at the end of the test.

Since an expert system is a tool applicable to various fields, we aim to reach a wide variety of users. Indeed, in the expert system that we developed, rules can be written by experts in fuzzy logic, by scientists who are not specialists in expert systems and fuzzy logic, but who want to integrate into a software or a more complex system. Moreover, our industrial partners may want to formalize their own knowledge or give the opportunity to their end users to configure their system.

We tried to have a panel which reflect this variety of users: it consists in 27 varied users (52 % men and 48 % women), age 20-57. Among these users, we selected six expert users (who use fuzzy expert systems in their business), 8 scientists (who are not familiar with fuzzy expert systems) and 13 non-scientific users. All these users are not the partners we interviewed before.

The application domain of the experiment is home automation. Users are invited to author rules to control automatic shutters according to various observed criteria: the presence of the person in the room, the brightness, the wind and the indoor and outdoor temperatures. The test was divided into three exercises that will be resolved by each of three interfaces:

- The first exercise consists in typing rules based on two inputs only. No help has been given to the user in the use of the three interfaces. The goal is to master the different tools. The time of completion of this exercise reflects in part the time of the interface handling.
- Between the first and the second exercise, the user who failed to master at least one interface receives explanations. Exercise 2 also involves writing rules based on two inputs. This time, the execution time reflects how quickly it is possible to write a rule.
- The third exercise consists in writing more complex rules that cannot be authored with the association matrices. The user is asked to account for this limitation alone.

In practice, the tests lasted between one and two hours, depending on the candidate. It explains partially why the panel remains restricted.

In order not to introduce biases in the assessment, two measures were taken. Firstly, the rules to write were given to the users by playing on the causal constructions of the natural language: for example, by employing a construction of the form "when the wind is strong, the shutters should be raised" instead of "If the wind is strong then the position of shutters is raised". This allows not to take into account the difficulty of the user to carry out logical reasoning. Secondly, it was decided to present the GUIs in random order so that no technique is discriminated.

Participants were asked to complete a very detailed questionnaire in order to identify factors that may explain their performance, but statistical tests showed that only the level of education has an impact. For example we asked people if they had a smartphone with touchscreen (in this case, *drag-and-drop* on touch surface would be under control), but 100 % of the panel have such a terminal.

4.2 Results

We performed statistical tests on the results obtained with the different exercises, both in terms of time and number of errors. However, considering the population and the amplitude of the results, Wilcoxon tests did not yield significant results. We then applied ANOVA (analysis of variance). For this purpose, we conducted tests of homoscedasticity variance (Bartlett's test) indicating that the variances between our classes are significantly homogeneous. Finally, ANOVA results indicate a significant difference between the three tools with a confidence level of 10 %.

Figure 11 indicates the durations of the exercise 1 per level of education (undergraduate or graduate). The figure shows different statistical elements:

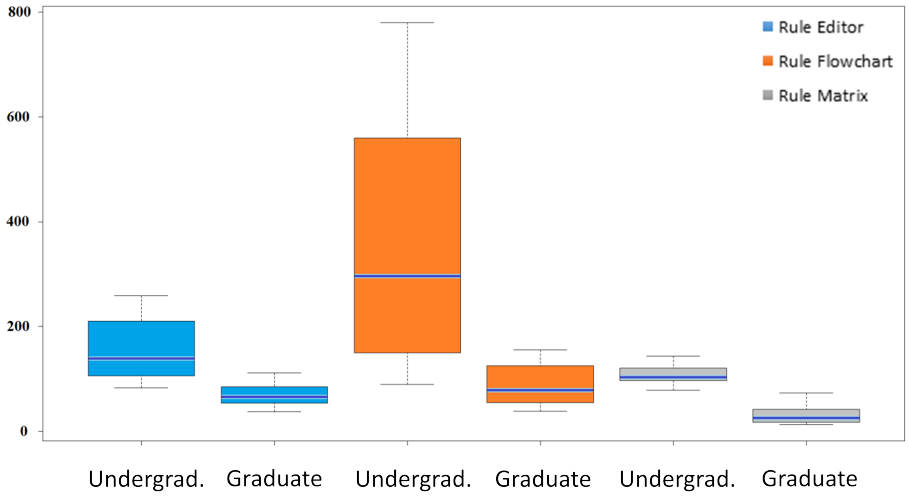

Figure 11: Duration (in s) of exercise 1 per education level (undergraduate or graduate).

the average is represented by a thick line contained in the box plot. The box extends from the first quartile to the third quartile and the whiskers extend to at least 1.5 times the inter-quartile range. The durations of exercise 1 are longer than the durations of the others because this exercise includes the discovery of the tools by the participant who does not receive any outside assistance at this stage of the experimentation. These graphs show firstly the disparity between undergraduate participants and graduate ones: the average duration is always longer. Then, the range of durations for undergraduate people is larger. In addition, the graphs show that flowchart is the most difficult tool to handle, especially by undergraduate people, while in contrast, everyone finishes quickly the exercise with association matrix. To understand these figures, we have to remember that there are few ways to interact with the matrix: users just have to choose two inputs and select a value in each cell. It is therefore normal that the tool is mastered quickly. Finally, the durations with our tool is longer than for the association matrix, although average durations are comparable. Undergraduate participants have also found more difficult to overcome *Rule editor* than *Rule matrix*.

The durations of the other exercises are quite similar except that the maximum durations are lower than those of exercise 1 (see table 1). The duration of exercise 3 with Rule matrix is just the time to understand it is not possible to use this GUI for the exercise. The association matrix remains the fastest tool to create rules, closely followed by our rule editor. The flowcharts are still slower, especially for undergraduate people (on average, there are more than 2 minutes between the two populations). In terms of population categories, experts in artificial intelligence have achieved the fastest all exercises regardless of the tool,

Exercise	GUI	Undergraduates	Graduates
Exercise 2	Rule Editor	0'58"	0'36"
	Rule Flowchart	1'34"	0'40"
	Rule matrix	2'44"	0'35"
Exercise 3	Rule Editor	3'09"	2'01"
	Rule Flowchart	4'52"	2'09"
	Rule matrix	0'48"	0'25"

Table 1: Average durations for exercise 2 and exercise 3 per education level.

followed by scientists and then by the other candidates. These results confirm the intuition.

In terms of speed of mastering and execution, nothing seems faster than the association matrix. However, for now we have considered only the execution time of the exercises without looking at the errors produced by users. Statistical tests show that the number of errors is independent of the education level.

Figure 12 shows the error rate of the whole panel on each exercise and for each tool. Errors are mostly logical misunderstanding (e.g. the use of a disjunction instead of a conjunction). Whereas association matrices are the fastest way to capture the rules, they get the highest error rate, which increases with the difficulty of the exercise. Browsing through input pairs to create the rules seems to make the exercise difficult and requires more concentration. We also count as error the rules that were created when the user try to write rules with more than two entries (which, we recall, is impossible). Finally, the rules are more difficult to read when displayed in a tabular form than a textual form. This is also the case for flowcharts, which could also explain the error rate of the *Rule flowchart* tool. However, with the use of the *Rule editor* tool, the users make fewer mistakes: this can be explained by the fact that the rules are presented in an easily readable form, unlike the other tools. In the case of *Rule editor*, only one mistake was committed.

On a more subjective way, we asked the participants what was the tool they would prefer if they had to use daily at home. All undergraduate participants answered they preferred *Rule editor*. This makes sense since our tool is not based on a mathematical or logical representation that these users are not used to. However, for graduate participants, the answers are a bit more varied: 43 % responded *Rule editor*, 29 % *Rule flowchart*, 9 % *Rule matrix* and finally 17 % would use *rule editor* for small and medium rules and *rule flowchart* for building more complex rules.

We also compared the answers regarding the level of knowledge in fuzzy expert systems (expert, intermediate, novice): 60 % of the experts prefer *Rule editor*, others are distributed equitably between the other two tools. No novice users have appreciated *Rule matrix*, despite what the quantitative analysis revealed; however 50 % of them chose *Rule editor* and 12 % a mix between *Rule editor* and *Rule flowchart*. Finally, 60 % of intermediate users chose *Rule editor*.

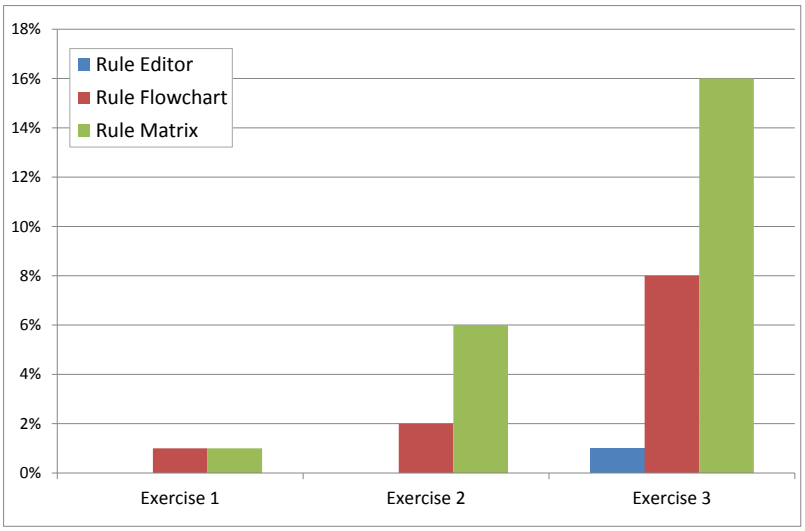

Figure 12: Error rate of all users, per exercise and per tool (in %).

	Experts	Intermediate	Novices
Rule editor	60%	60%	50%
Rule matrix	20%	10%	0%
Rule flowchart	20%	10%	38%
Rule editor + Rule flowchart	0%	20%	12%

Table 2: User preferences for a tool per skills.

All figures are reported in Table 2.

5 Discussion

According to the experimental results, association matrices are quickly and easily adopted by graduated people. However, for people less familiar with mathematics and logic, it seems to be less easy and the mistakes are numerous. In addition, the tool is very limited because only rules with two inputs and conjunctions can be drafted.

Similarly, scientific people appreciate flowcharts, especially for complex rules. However, the reading is not easy and novice users are struggling to reason correctly and check their rules.

The *Rule editor* tool seems to suit the majority of users. The adoption of the

tool and the rule authoring speed are similar to the association matrices. On one hand, textual form helps to achieve a low error rate. On the other hand, the time to author a rule is slightly greater than with association matrices (if the rule can be supported by matrices). Moreover, the questionnaire helped us to realize that the icons representing operators (see Figure 5) were not explicit for most of them and had to provide context-sensitive help directly on the graphical interface for even faster handling.

The principle of interactions proposed in this paper is applied, without loss of generality, to fuzzy expert systems: in fact, it also applies to the other expert systems, including business rule management systems, often used in large companies. Other industrial systems also use knowledge including constraints solvers, which are used for example for resource scheduling, or optimization engines, whose constraints are expressed by arithmetic and logical formulas.

6 Conclusion

In this paper, we proposed a new tool for knowledge acquisition. We illustrated this tool with the authoring of rules for a fuzzy expert system. Our inspiration comes from the various discussions with our industrial partners. Modern interaction techniques are exploited to provide an enjoyable user experience, and fast and intuitive mastering: tablets are ideal for use *drag-and-drop* which is a popular user interaction, and the majority of users are familiar with it.

We conducted a test campaign on 27 users with different profiles to compare our tool with the two major tools in the field of fuzzy expert systems. Tests show that a majority of users appreciate the handling and use of our graphical interface and that it allows to reduce the error rate.

Confronting the graphical interface to users enabled us to introduce new features and add context-sensitive help when a drag operation begins or when a click or a touch is detected. This GUI is now a productivity tool we share with our partners, and has been used to capture up to 299 rules in a row.

Acknowledgment

This research has been supported by BPI-France with the EDENS project (Eco-District Energy Network Systems). We want to thank Bastien Guillon for helping us maintaining the software.

References

[1] T. Winograd, "Shifting viewpoints: Artificial intelligence and human-computer interaction," *Artificial Intelligence*, vol. 170, pp. 1256–1258, 2006.

[2] Jameson, Anthony, Spaulding, Aaron, Yorke-Smith, and Neil, "Introduction to the special issue on Usable AI," *AI Magazine*, vol. 30, no. 4, 2009.

[3] K. Z. Gajos and D. S. Weld, "Usable ai: Experience and reflections," in *CHI'08 Workshop on Usable Artificial Intelligence*, April 2008.

[4] A.-S. Dadzie, V. Lanfranchi, and D. Petrelli, "Towards ai usability: Problems, strategies and practicals," in *CHI'08 extended abstracts on Human Factors in Computing Systems*, 2008.

[5] N. Caprani, N. E. O Connor, and C. Gurrin, "Touch screens for the older use," in *Assistive Technologies*. Fernando A. Auat Cheein, 2012.

[6] L. Motti, L. Vigouroux, and P. Gorce, "Drag-and-drop for older adults using touchscreen devices: effects of screen sizes and interaction techniques on accuracy," *Proceedings of the 26th Conference francophone on Interaction Homme-Machine (IHM'14)*, pp. 139–146, 2014.

[7] K. Jones, "Natural language processing: A historical review," in *Current Issues in Computational Linguistics: In Honour of Don Walker*, ser. Linguistica Computazionale, A. Zampolli, N. Calzolari, and M. Palmer, Eds. Springer Netherlands, 1994, vol. 9, pp. 3–16.

[8] M. Nii, T. Yamaguchi, Y. Takahashi, A. Uchinuno, and R. Sakashita, "Fuzzy rule extraction from nursing-care texts," in *Multiple-Valued Logic, 2009. ISMVL '09. 39th International Symposium on*, May 2009, pp. 30–35.

[9] S. Hassanpour, M. O'Connor, and A. Das, "A framework for the automatic extraction of rules from online text," in *Rule-Based Reasoning, Programming, and Applications*, ser. Lecture Notes in Computer Science, N. Bassiliades, G. Governatori, and A. Paschke, Eds. Springer Berlin Heidelberg, 2011, vol. 6826, pp. 266–280.

[10] J. Darragh, I. H. Witten, and M. James, "The reactive keyboard: a predictive typing aid," *Computer*, vol. 23, no. 11, pp. 41–49, Nov 1990.

[11] G. Acampora and V. Loia, "Fuzzy markup language: A new solution for transparent intelligent agents," in *Intelligent Agent (IA), 2011 IEEE Symposium on*, April 2011, pp. 1–6.

[12] N. Henze, E. Rukzio, and S. Boll, "Observational and experimental investigation of typing behaviour using virtual keyboards for mobile devices," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '12. New York, NY, USA: ACM, 2012, pp. 2659–2668.

[13] M. Mosconi and M. Porta, "A data-flow visual approach to symbolic computing: implementing a production-rule-based programming system through a general-purpose data-flow vl," in *Visual Languages, 2000. Proceedings. 2000 IEEE International Symposium on*, 2000, pp. 83–84.

[14] M. Chein, M. Mugnier, and M. Croitoru, "Visual reasoning with graph-based mechanisms: the good, the better and the best," *The knowledge engineering review*, no. 28, pp. 28–51, 2013.

[15] M. Smith, E. Black-Ziegelbein, and T. Hee, "Drag and drop rule topology," U.S. Patent 8,140,991, March 20, 2012.

[16] M. Neale, "Natural language enhanced user interface in a business rule management," US Patent 8,364,469, 01 29, 2013.

[17] E. Fontaine, A. Demeure, J. Coutaz, and N. Mandran, "Retour d'expérience sur kiss, un outil de développement d'habitat intelligent par l'utilisateur final," in *Proceedings of the 2012 Conference on Ergonomie Et Interaction Homme-machine*, ser. Ergo'IHM '12. New York, NY, USA: ACM, 2012, pp. 153:153–153:160.

[18] J. Coutaz, A. Demeure, S. Caffiau, and J. L. Crowley, "Early lessons from the development of spok, an end-user development environment for smart homes," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, ser. UbiComp '14 Adjunct. New York, NY, USA: ACM, 2014, pp. 895–902.

[19] G. H. Subramanian, J. Nosek, S. P. Raghunathan, and S. S. Kanitkar, "A comparison of the decision table and tree," *Commun. ACM*, vol. 35, no. 1, pp. 89–94, Jan. 1992.

[20] I. Vessey and R. Weber, "Structured tools and conditional logic: An empirical investigation," *Commun. ACM*, vol. 29, no. 1, pp. 48–57, Jan. 1986.