Generation of Similarity Measures from Different Sources

Benno Stein and Oliver Niggemann

Dept. of Mathematics and Computer Science—Knowledge-Based Systems,
University of Paderborn, D–33095 Paderborn, Germany
{stein,murray}@uni-paderborn.de

Abstract Knowledge that quantifies the similarity between complex objects forms a vital part of problem-solving expertise within several knowledge-intensive tasks. This paper shows how implicit knowledge about object similarities is made explicit in the form of a similarity measure.
The development of a similarity measure is highly domain-dependent. We will use the domain of fluidic engineering as a complex and realistic platform to present our ideas. The evaluation of the similarity between two fluidic circuits is needed for several tasks: (*i*) Design problems can be supported by retrieving an existing circuit which resembles an (incomplete) circuit description. (*ii*) The problem of visualizing technical documents can be reduced to the problem of arranging similar documents with respect to their similarity.
The paper in hand presents new approaches for the construction of a similarity function: Based on knowledge sources that allow for an expert-friendly knowledge acquisition, machine learning is used to compute an explicit similarity function from the acquainted knowledge.
Keywords. Machine Learning, Knowledge Acquisition, Case-Based Reasoning

1 Introduction

This paper addresses a key aspect within various knowledge-based analysis and synthesis tasks: The construction of a measure that adequately models the similarity between two problem instances. This may be the similarity between two documents within a document retrieval task, the similarity between two cases within a case-based reasoning task, or a similarity assessment between two points in a graph when working on a visualization task.

When given two problem instances, a domain expert is in a position to assess the similarity between these instances with respect to a problem solving task in hand. It is a question of high importance, and it is the central question of this paper how this part of an expert's problem-solving expertise can be elicited and made explicit.

In its general form, a set of objects (problem instances), O, is given, where each object is described by a set of features $x_1, \ldots x_n$. The similarity between two objects x and y is taken to assess the usability of a solution of instance x as a solution for instance y. Usability can be stated a-posteriori only while the similarity between two objects can be stated immediately [14]. The quantification of the concept usability by means of the similarity between two feature vectors shows the crucial importance that comes up to the computation of the features.

In the following, the similarity between two objects x and y is designated by a relation "sim" where $sim(x, y)$ determines a value from the interval $[0; 1]$. The larger is the value of sim the more similar are x and y to each other.

1.1 Where Similarity Measures Come from

A similarity measure establishes a particular form of knowledge, which—using AI terminology—can be acquainted from some source. An often applied concept to elicit similarity knowledge is the interview of domain experts: "Are these two problem instances, x and y, similar?" "What are the significant features that make x and y similar?" "To which extent are x and y similar?"

These sample questions make problems of the concept "knowledge acquisition by questioning" obvious. On the one hand, it is hard for the domain expert to give quantifiable answers, while on the other hand, it is hard for the knowledge engineer to access the quality of these answers.

Note that a similarity measure can also be constructed from other knowledge sources; for instance from the knowledge that is encoded within an existing object classification [21]. Obviously, each classification implies knowledge about feature relevance and feature similarity with respect to the classified objects. Given such a knowledge source, methods from the field of machine learning can be used to transform implicit knowledge on object similarities into an explicit similarity measure.

In Section 4, we will concentrate on three different knowledge sources where classification knowledge can appear in:

1. *Partial Similarity Quantification.* Similarity assessments are known for only a subset of $O \times O$, which is called the learning set here.
2. *Partitioning of O.* The knowledge on object similarity is encoded through the division of (a subset of) O into equivalence classes respecting particular similarity aspects.
3. *Graphical Similarity Specification.* The knowledge on object similarity is encoded within geometric distances of a 2-dimensional plot.

The development of a similarity measure is highly domain dependent. Hence we will introduce a particular domain from which the object set O is drawn and which will serve as realistic example throughout the paper. This domain is the domain of fluidic engineering.

2 Using Similarity Measures in Fluidic Engineering

Similarity measures can be used for those tasks in fluidic engineering that are not treated at a deep, physical level of behavior but at the much more abstract level of function. At this level, the complex physics of a fluidic circuit is reduced to a set of features which characterizes the circuit's usability to fulfill a desired function. The following list outlines tasks that are solved at an abstract functional level.

- *Functional Analysis.* Check whether two fluidic systems are similar with respect to their intended operation [22].
- *Fluidic System Design.* Construct a new fluidic system by coupling together already designed units (fluidic axes) from different systems [20,9].
- *Document Retrieval.* Query a database for diagrams of fluidic systems that are similar with respect to a given set of demands.

The model of fluidic function as specified in Definition 1 establishes the knowledge level at which the mentioned tasks are solved. Subsection 3.2 shows in which way this functional model is encoded as a feature vector for fluidic circuit objects.

Taken an engineer's point of view, the gist of a model of fluidic function consists of a set of state variables, F_X, along with the discrete state prescription function, Δ. Each state variable in F_X represents a subfunction of a fluidic axis; Δ characterizes the behavior of the fluidic axes by means of the working phases of the output units, which are cylinders in most cases.

Definition 1 (Model of Fluidic Function). *Let S be a fluidic system. A model of fluidic function of S is a discrete event model $\langle F_X, F_Y, \mathcal{X}, \mathcal{Y}, \Delta, \Lambda \rangle$ whose elements are defined as follows.*

1. *F_X is the set of state variables. Each state variable corresponds one-to-one to a fluidic axis in S and defines the phases that can be adopted by this axis. Hence, $|F_X|$ is the total number of axes in S. F_Y is the set of output variables, defining the positions, the velocities, and the pressures at the working elements within the fluidic axes.*
2. *The sets X_f and Y_f designate the domains of the variables f in F_X and F_Y respectively. Likewise, $\mathcal{X}$ designate the Cartesian product of the state variable domains, and $\mathcal{Y}$ designate the Cartesian product of the output variable domains.*
3. *$\Delta : \mathbf{R}^+ \rightarrow \mathcal{X}$ is the discrete state prescription function and specifies the phase transitions of a model of fluidic function. Given a point in time, $t \in \mathbf{R}^+$, Δ determines a vector of phases.*
4. *$\Lambda : \mathcal{X} \times \mathbf{R}^+ \rightarrow \mathcal{Y}$ is the output function. Given a vector of phases, $\mathbf{x} \in \mathcal{X}$, and a point in time, $t \in \mathbf{R}^+$, Λ determines an output vector, $\mathbf{y} \in \mathcal{Y}$.*

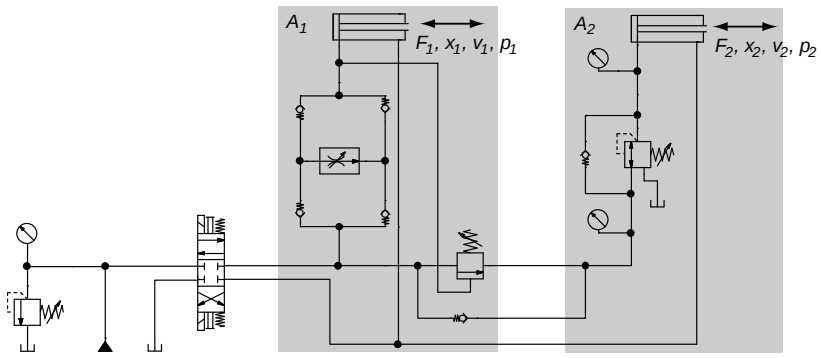


Figure 1. Hydraulic circuit with two axes A_1, A_2. $F_1, F_2, x_1, x_2, v_1, v_2, p_1$, and p_2 designate the forces, positions, velocities, and pressures respectively that are necessary to define a functional model of the circuit.

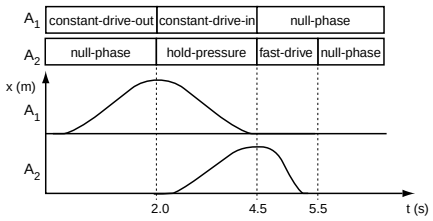
Example. The example specifies the functional model of the fluidic system in Figure 1. The shown circuit contains two axes each of which having three states, say, phases.

1. *Variables.* $F_X = \{P_1, P_2\}$, $F_Y = \{x_1, x_2, v_1, v_2, p_1, p_2\}$, $F = F_X \cup F_Y$.

2. *Domains.* $X_{P_1} = X_{P_2} = \{$*constant-drive-in, constant-drive-out, fast-drive, null-phase, hold-pressure*$\}$, $Y_f = \mathbf{R}^+,\ \ f \in \{x_1, x_2, p_1, p_2\}$, $Y_{v_1} = Y_{v_2} = \mathbf{R}$.
3. *State Prescription.* $\Delta : \mathbf{R}^+ \rightarrow X_{P_1} \times X_{P_2}$. Given an absolute point in time t, the function Δ is given in Table 1 (left-hand side).

Table 1. State prescription (left-hand side) and output function (position-time diagrams, right-hand side) of the hydraulic axes, A_1, A_2.

t	P_1	P_2
$[0; 1.6)$	*constant-drive-out*	*null-phase*
$[1.6; 3.2)$	*constant-drive-in*	*hold-pressure*
$[3.2; 4.0)$	*null-phase*	*fast-drive*
≥ 4.0	*null-phase*	*null-phase*



4. *Output Function.* Typically, the output function is represented by means of different graphs which show the courses of the positions, velocities, or pressures at the working elements for a particular input. The right-hand side of Table 1 shows the phase transitions and position-time diagrams for a particular input.

Remarks. The abstraction from a physical behavior model, which is grounded on continuous time, towards a functional model, which is event-based, can be done automatically. For this, the state prescription function of the continuous time model is simulated and investigated with respect to intervals of stationary velocity. However, the development of such an abstraction is not discussed in this place.

3 Learning Similarity Measures

Since any automatic comparison between fluidic drawings themself is hardly possible, an abstract circuit description has to be found. The functional model developed in Section 2 captures the abstract aspects of fluidic drawings that are also used by an expert to assess circuit similarities.

The subject of this section is the application of learning methods to the construction of similarity measures. These algorithms bridge the gap between implicit expert knowledge about circuit similarities and explicit similarity functions.

In order to apply machine learning, the functional model has to be encoded into a so-called feature vector. A feature vector is a list of values describing an system. For instance, $\langle$Smith, James, 1967, 85 kg$\rangle$ is a feature vector describing a person.

3.1 On Similarity Measures in General

Much work has been done in the last decades on similarity measures; good overviews can be found in [16,7,12,23,15]. A common similarity measure is the simple weighted linear similarity function. Let $(f_1^{(1)}, \ldots, f_p^{(1)}), (f_1^{(2)}, \ldots, f_p^{(2)}), p \in \mathbf{N}$ be two feature vectors. Then the simple weighted linear similarity measure is defined as:

$$\sum_{1 \leq i \leq p} w_i \cdot (f_i^{(1)} \ominus f_i^{(2)}), w_i \in \mathbf{R}$$

The definition of the operator $\ominus$ depends on the feature's type:

- *Cardinal.* We call a features cardinal if and only if all values of the feature are real numbers. Typical cardinal features are height, temperature, or distances. Values of cardinal features can be added, subtracted, multiplied, and divided and the result is still a reasonable value for the feature. For cardinal features, $x \ominus y$ is normally defined as $|x - y|$, i. e. the simple weighted linear similarity measure can be rewritten as $\sum_{1 \leq i \leq p} w_i \cdot |f_i^{(1)} - f_i^{(2)}|, w_i \in \mathbf{R}$.
- *Nominal.* Nominal values can only be compared with respect to equality. Name or profession of a person are nominal features. If a nominal feature has only two possible values (e. g. gender of a person), it is called a binary feature. For nominal features, x, y, the following definition is often used [24]:

$$x \ominus y = \begin{cases} 0, \text{ if } x = y \\ 1, \text{ otherwise} \end{cases}$$

Note that learning such functions means finding values for the parameters w_i. More complex distance functions have been examined by the authors in [21].

3.2 A Similarity Measure for Fluidic Systems

As described in Section 2, each fluidic system S can be reduced to a functional model. In accordance with [9,22], the similarity between two hydraulic systems, S_1, S_2, is defined using the similarities between individual axes in the respective circuits:

$$sim(S_1, S_2) = \sum_{A_1 \in \mathcal{A}_1} \max\{sim_{axes}(A_1, A_2) \mid A_2 \in \mathcal{A}_2\}$$

where $\mathcal{A}_1$ denotes the axes of S_1, $\mathcal{A}_2$ denotes the axes of S_2, sim_{axes} denotes a function measuring the similarity between two axes, and $|\mathcal{A}_1| \leq |\mathcal{A}_2|$ holds.

For two axes, A_1, A_2, the similarity function $sim_{axes}(A_1, A_2)$ is defined by the function $sim(\boldsymbol{f}(A_1), \boldsymbol{f}(A_2))$ where $\boldsymbol{f}(A_i)$ denotes a vector of cardinal features describing A_i, i. e. $\boldsymbol{f}(A_i) \in \mathbf{R}^m, i = 1, 2$. In the following text, the simple weighted linear similarity function $sim(\boldsymbol{f}(A_1), \boldsymbol{f}(A_2)) = \sum_{i=1}^{m} w_i \cdot |f_i^1 - f_i^2|$ is used.

The feature vector of fluidic axes, $\boldsymbol{f}(A)$, which is necessary to compute the similarity between two fluidic circuits, S_1, S_2, can directly be extracted from the functional models of S_1 and S_2. Each axis is described by two types of features: (*i*) phase descriptions and (*ii*) phase orders. These groups of features are defined as follows.

(*i*) *Phase Descriptions.* Phases are classified into the categories *constant-drive, position-drive, hold-position, accelerate, fast-drive, hold-pressure,* and *press*; each category in turn is characterized by 6 features:

1. How many phases of the specific category exist in the respective axis?
2. How long (in seconds) is the duration of the phase ?
3. Which distance (in mm) is covered by the working element?
4. Which force (in Newton) is applied to the working element?
5. How precisely must the axis work? This is a value from $[0; 1]$ that defines the acceptable deviations from the duration, the distance, and the force.

(*ii*) *Phase Orders.* These 49 features $f^{\text{ord}}_{i,j}, 1 \leq i,j \leq 7$ capture the order of the phases. For example, $f^{\text{ord}}_{1,2}$ is the number of times a phase of category 1 (*constant-drive*) is directly followed by a phase of category 2 (*position-drive*).

Together, the feature vector $\boldsymbol{f}(A)$ for an axis A is of the following form:

$$f(A) = \big(\ (\text{phase description } \textit{constant-drive}), (\text{phase description } \textit{position-drive}), (\text{phase description } \textit{hold-position}), (\text{phase description } \textit{accelerate}), (\text{phase description } \textit{fast-drive}), (\text{phase description } \textit{hold-pressure}), (\text{phase description } \textit{press}), (\text{phase orders})\ \big)^T$$

Related to the example from Page 3, the feature vectors for the axes A_1 and A_2 are given as follows.

$$f(A_1) = \big(\ \underbrace{(2, 1.6, 500, 300, 0.8), (), (), (), (), (), ()}_{\text{phase descriptions}}, \underbrace{(1, 0, \ldots, 0)}_{\text{phase orders}}\ \big)^T$$

$$f(A_2) = \big(\ \overbrace{(), (), (), (), (1, 0.8, 500, 0, 0.6), (1, 1.6, 500, 2000, 0.8), ()}, \underbrace{(0, \ldots, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)}_{\text{phase orders}}\ \big)^T$$

3.3 Methods for Constructing Similarity Measures

Existing methods for constructing similarity measures can be divided into two main classes: (*i*) Methods that employ reinforcement-learning and (*ii*) algorithms that rely on statistical analysis for the main part.

Reinforcement-learning methods predict a similarity value and ask the user or a different system to rate the prediction. Based on this rating the weights w_i are adjusted. Statistical methods analyze given examples and deduce appropriate weights. Table 2 lists representatives of well known methods; more examples can be found in [4,1,19].

Table 2. Selected existing methods for the construction of similarity measures.

Name	Type	Remarks	Literature
EACH	reinforcement-learning	extra parameters needed	[17]
RELIEF	reinforcement-learning	binary weights	[13]
CCF	statistical	only binary features	[6]
GM-CDW	statistical		[11]

These methods have in common that the knowledge acquisition step (how to obtain the necessary knowledge from an expert) and the learning step (finding appropriate values for the weights w_i) are not treated separately. Our approach, which is described in the next section, differentiates between these two steps.

Combining the knowledge acquisition step and the learning step entails several problems:

– Since the expert is integrated into such methods in a predefined manner, no flexibility is granted in the way the knowledge is obtained. Hence additional knowledge sources cannot be tapped.

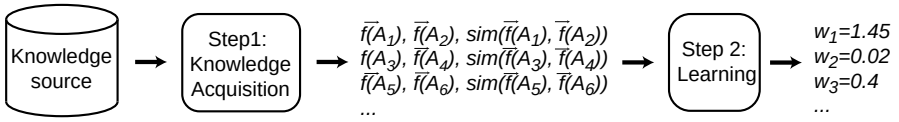


Figure 2. The general framework for the construction of similarity measures.

- Although the methods mentioned in Table 2 rely on standard learning paradigms such as reinforcement learning, they do not apply standard learning algorithms such as regression or neural networks but employ proprietary algorithms. While for standard learning algorithms advantages and disadvantages have been examined, almost nothing is known about the proprietary algorithms.
- Verifying a combined method is difficult since learning problems cannot be distinguished from knowledge acquisition problems.

3.4 A General Framework

Figure 2 shows the general framework for constructing similarity measures used in this paper: Two main steps are identified, a knowledge acquisition step and a learning step. This separation allows for the usage of both different acquisition methods and different learning methods to obtain the necessary information from an expert. The first step always results in a set of feature-vector-pairs whose similarity is known (see Figure 2), e. g. $\{(\boldsymbol{A}_1, \boldsymbol{A}_2, sim(\boldsymbol{f}(A_1), \boldsymbol{f}(A_2))), (\boldsymbol{A}_3, \boldsymbol{A}_4, sim(\boldsymbol{f}(A_3), \boldsymbol{f}(A_4))), \ldots\}$

The second step uses the rated vector pairs and applies a supervised learning strategy to find values for the weights w_i. For our applications both regression and neural networks have been used. Note that only a small but typical set of objects, the learning set, is used for learning purposes.

4 Knowledge Acquisition

In Section 3, the definition of similarities between fluidic circuit has been reduced to the problem of finding similarities between fluidic axes. Finding such similarities poses a knowledge acquisition problem, because experts have the necessary knowledge only implicitly. When asked to express their knowledge about axes similarities explicitly as a mathematical function, most experts are overstrained.

Below, three methods that tackle the knowledge acquisition problem are presented. The methods differ from each other in the explicitness of the underlying knowledge source.

4.1 Knowledge Source 1: Partial Similarity Quantification

This technique is quite simple: The expert assesses the similarity of m axes pairs which are presented to him, e. g. on a scale from 0 to 10. These assessments are used to learn a similarity measure which then is used to compute the similarity between n, $n \gg m$, pairs of axes.

Exploiting this knowledge source has disadvantages:

- If m axes are used to build the measure, the expert has to rate $\frac{m^2}{2} - m$ object pairs.
- The similarities have to be comparable; i. e., the expert has to decide whether object A is more similar to object B then object C to D. Such decisions turn out to be quite difficult.

In the field of statistics, the problem of assessing object similarity is well known. Most methods,[1] such as ranking, free sorting, or anchor stimulus, rely on a user who defines all object similarities. Hence, no learning or abstraction mechanisms are applied.

4.2 Knowledge Source 2: Partitioning the Set of Objects

For this method the expert has to classify the axes. Two axes are similar if and only if they belong to the same class. Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ be the set of axes and let $c : \mathcal{A} \to \mathcal{C}$ be the classification function, where $\mathcal{C}$ comprises the set of possible classes. The classification function c is specified by the domain expert. Then the similarity sim is defined as follows.

$$\forall_{A_1,A_2,\in\mathcal{A}} : \; sim(\boldsymbol{f}(A_1), \boldsymbol{f}(A_2)) = \begin{cases} 1, \text{ if } c(A_1) = c(A_2) \\ 0, \text{ otherwise} \end{cases}$$

Reasonable classes, for example, are $\mathcal{C} = \{$*manipulation, press, hold*$\}$ or $\mathcal{C} = \{$*high-pressure, low-pressure, fast-drive*$\}$.

The main disadvantage bound up with this knowledge source is that a partitioning, say, a disjunctive classification, is sometimes difficult to be stated. The advantages of this knowledge source are:

- n classifications define $\frac{n^2}{2}$ similarities.
- Domain experts have few problems in classifying fluidic axes.

Although in the learning set only the similarity values 0 and 1 are given, learning algorithms like regression result in similarity measures that can yield any similarity value from the interval $[0; 1]$. This is because the given data is abstracted by the learning algorithms.

4.3 Knowledge Source 3: Graphical Similarity Specification

The method presented now demands a minimum of explicit knowledge. The expert is asked for an exemplary visualization of his understanding of the similarity between objects. By means of a computer, this visualization is abstracted towards a graph, from which a similarity measure is computed. No additional knowledge is demanded from the expert, i. e., this method can always be applied.

Again let $\mathcal{A} = \{A_1, \ldots, A_n\}$ be the set of axes. The expert manually defines a layout by specifying a function $\rho : \mathcal{A} \to \mathbf{N} \times \mathbf{N}$, which defines a two-dimensional position for each axis. The similarity of two axes A_1, A_2 is defined by:

$$sim(\boldsymbol{f}(A_1), \boldsymbol{f}(A_2)) = -||A_1, A_2||_2$$

where $||x, y||_2$ denotes the Euclidean distance between the positions of x and y.

[1] Good overviews can be found in [2,5].

The following points distinguish this knowledge source from the previous one:

- The graphical definition of similarities is closer to the mental model of the user than is the definition of a classification function c.
- By placing n objects, $\frac{n^2}{2}$ similarities are defined.

A difficulty of the graphical similarity specification is that by placing one axis, the distances to $n-1$ other objects must be taken into account. To simplify this layout problem, only object distances up to certain maximum distance are considered. For this, the layout is clustered in a first step, say, groups of closely related objects are identified.

5 Learning and Results

Input for the learning step (cf. Figure 2) was a set of rated axes pairs $\{(\boldsymbol{A}_1, \boldsymbol{A}_2, sim(\boldsymbol{f}(A_1), \boldsymbol{f}(A_2))), (\boldsymbol{A}_3, \boldsymbol{A}_4, sim(\boldsymbol{f}(A_3), \boldsymbol{f}(A_4))), \ldots\}$, which was used to find values for the weights w_i of the similarity function $sim(\boldsymbol{f}(A_1), \boldsymbol{f}(A_2)) = \sum_{i=1}^{m} w_i \cdot |f_i^1 - f_i^2|$. Learning was done by applying least-square regression and by means of neural networks. Details can be found in [3,18,25,10].

The acquisition methods described in Subsection 4.2 (knowledge source 2) and 4.3 (knowledge source 3) have been implemented and applied to the learning of similarity measures for fluidic axes.

- *Knowledge Source 2.* 67 fluidic axes were classified into 9 classes by a domain expert. Using the method described on Page 8, a similarity measure was constructed. The error rate was defined as the percentage of axes pairs whose similarity was assessed incorrectly by the learned measure. The error rate on the learning set[2] was 12%, while the error rate on a test set[3] was 16%. Obviously, a good similarity measure has been constructed.
- *Knowledge Source 3.* A graphical arrangement of circuit documents, which has been proposed by the domain expert, was analyzed and similarity measures were constructed. To evaluate the quality of the learned measures the Mean Square Error (MSE) was used:

$$\sqrt{\sum_{A_i, A_j} (sim^t(A_i, A_j) - sim^e(A_i, A_j))^2},$$

  A_i and A_j denote fluidic axes, sim^t denotes the similarity as predicted by the learned similarity measure, and sim^e denotes the empirical similarity measure defined by the manual layout. On a test set, an MSE of 0.22 has been achieved. All similarity values are from the interval $[0, 1]$, hence the MSE defines an average variation from the correct similarity. I. e., also with this knowledge source a good similarity measure could be constructed.

[2] The learning set comprised axes pairs used for the learning process.

[3] The axes pairs in the test set have not been used for the learning process.

References

1. D. Aha. Tolerating noisy, irrelevant, and novel attributes in instance-based learning algorithms. *International Journal of Man-Machine Studies*, 1992.
2. K. Backhaus, B. Erichson, W. Plinke, and R. Weber. *Multiv. Anaylsem.*. Springer, 1996.
3. R. Beale and T. Jackson. *Neural Computing*. Inst. of Physics, Bristol, Phil., 1994.
4. A. Bonzano, P. Cunningham, and B. Smyth. Using introspective learning to improve retrieval in cbr: A case study in air traffic control. In *Second ICCBR Conf.*, 1997.
5. I. Borg and P. Groenen. *Modern Multidimensional Scaling*. Springer, 1997.
6. R. H. Creecy, B. M. Masand, S. Smith, and D. Waltz. Trading mips and memory for knowledge engineering. *Communications of the ACM*, 35, 1992.
7. B. S. Everitt. Cluster analysis. Eward Arnolds, New York, Toronto, 1993.
8. F. Hayes-Roth, D. Waterman, and D. Lenat. *Building Expert Systems*. Addison Wesley Publishing Company, London, 1983.
9. M. Hoffmann. *Zur Automatisierung des Designprozesses fluidischer Systeme*. Diss., Univ. of Paderborn, Dept. of Mathematics and Computer Science, 1999.
10. D. W. Hosmer and S. Lemeshow. *Applied Logistic Regression*. Wiley & Sons, NY, 1989.
11. N. Howe and C. Cardie. Examining locally varying weights for nearest neighbor algorithms. In *Proceedings of the Eleventh ICML*. Morgan Kaufmann, 1997.
12. M. Jambu. *Explorative Datenanalyse*. Gustav Fischer Verlag, 1992.
13. K. Kira and L. Rendell. A practical approach to feature selection. In *Proceedings of the Ninth International Conference on Machine Learning*, 1992.
14. B. Nebel. Plan Modification versus Plan Generation. In A. Horz, editor, *7. Workshop "Planen und Konfigurieren", Hamburg*, number 723 in Arbeitspapiere der GMD, 1993.
15. A. Reckmann. Ähnlichkeitsmaße und deren Parametrisierung für die fallbasierte Diagnose am Beispiel einer medizinischen Anwendung. Master's thesis, Univ. of Paderborn, 1999.
16. M. M. Richter. Introduction to CBR. In M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard, and S. Weß, editors, *Case-Based Reasoning Technology. From Foundations to Applications*, Lecture Notes in Artificial Intelligence 1400, pages 1–15. Berlin: Springer-Verlag, 1998.
17. S. L. Salzberg. A nearest hyperrectangle learning method. *Machine Learning*, 1991.
18. W. S. Sarle. Neural Networks and Statistical Models. In *9th Annual SAS Users Group Intl. Conf.*, 1994. SAS Instit. Inc.
19. C. Stanfill and D. Waltz. Toward memory-based learning. *Communications of the ACM*, 29:1213–1228, 1986.
20. B. Stein. Optimized Design of Fluidic Drives—Objectives and Concepts. Techn. Rep. tr-ri-97-189, Uni. of Paderborn, Depart. of Mathematics and Computer Science, 1996.
21. B. Stein, O. Niggemann, and U. Husemeier. Learning Complex Similarity Measures. In *Jahrestagung der Gesellschaft für Klassifikation*, Bielefeld, Germany, 1999.
22. E. Vier. *Automatisierter Entwurf geregelter Hydrostatischer Systeme*, volume 795 of *Fortschritt-Berichte VDI. Reihe 8*. VDI, Düsseldorf, 1999.
23. S. Wess. Fallbasiertes Problemlösen in wissensbasierten Systemen zur Entscheidungsunterstützung und Diagnostik: Grundlagen, Systeme und Anwendungen. Technical report, Sankt Augustin: Infix, 1996.
24. D. Wilson and T. Martinez. Improved Heterogeneous Distance Functions. *Journal of Artificial Intelligence Research*, 6, 1997.
25. T. Wonnacott and R. Wonnacott. Regression: a second course in statistics. John Wiley & Sons, New York, Chichester/Brisbane/Toronto, 1981.