# The Knowledge Contained in Similarity Measures

Michael M. Richter, Kaiserslautern

Centre for Learning Systems & Applications

University of Kaiserslautern

# The Questions:

- Of which nature is the knowledge a similarity measure can contain?
- How to bring the knowledge into the measure?
- How to retrieve and use the knowledge for actual problems?

# The Relational Approach

## Basic Relations:

1) $R(x,y,u,v)$:
   "x and y are at least as similar as u and v are"

2) $S(z,x,y) :\Leftrightarrow R(z,x,z,y)$
   "z and x are at least as similar as z and y are"

3) $NN(z,x) :\Leftrightarrow \forall y\ S(z,x,y)$
   "x is a nearest neighbour of z"

# On the Semantics of Similarity-Measures

**Task:** Classification ($a \in U$, $b \in CB$)

**A plausible request:**

sim(a,b) = Prob(class(a) = class(b) | given observations)

Conditional Probability!

**Advantage:**

The Nearest-Neighbour-Principle is reduced to the Maximum-Likelihood-Principle

**Problem:**

What to do if we have very few observations and no other (a priori) information?

# Two Possible Approaches:

①

The Evidence-Approach (Dempster - Shafer):

Determine an evidence measure $\mu$ on the case base CB $\subseteq$ U,
(i.e. a probability on the power set of CB) ($a \in U$)

$$\mu_a : \wp(CB) \rightarrow [0,1]$$

Evidence measures reflect ignorance!

②

The Interval-Approach (Pöhlmann - Weichselberger):

Determine an interval for the (unknown) probability distribution:

$$I: U \times CB \rightarrow [0,1] \times [0,1]$$

$$I(a,b) = (x,y) \Rightarrow x \leq y$$

$x \leq \text{Prob}(\text{class}(a) = \text{class}(b) \mid \text{given observations}) \leq y$

The intervals also reflect ignorance!

# The Distribution of Knowledge in a CBR-System

Knowledge Sources

| Vocabulary<br>Attributes<br>Predicates... | Similarity<br>Measure<br>Sim | Selected Cases<br><br>Case Base CB |
|---|---|---|

Compiled Knowledge

$\Downarrow$

To be Interpreted at Run Time

$\Downarrow$

Compile Time:

Every Time before Actual Problem Solving

# Distribution of Knowledge

In principle, all knowledge could be

- in the case base:
  pure interpreter approach
  - all possible cases in CB = U

- in the measure:
  pure compiler approach

$$\text{sim}(a,b) = \begin{cases} 1 & \text{a and b are in the same class} \\ 0 & \text{otherwise} \end{cases}$$

# A Simple View on the Task of a CBR-System

Two simple tasks:

(1) Compute a function f(x)
(2) Decide for a,b $\in$ dom(f) : f(a) = f(b) ?

Observations:

- The ability to solve task (1) is sufficient for solving task (2)
- Task (2) may be a lot easier than task(1) e.g. $f(x) = x^2$
- Task (2) suffices for task (1) if a table

  $$(a_1, f(a_1)), (a_2, f(a_2)), \ldots$$

  for many $a_i$ is available

# Issues

- The Semantical Issue:
  What is the precise semantics of the parts of a CBR-system which can carry knowledge ?

- The Software-(Knowledge-)Engineering Issue:
  How is the transformation process
  Knowledge Sources → CBR-System
  best organized? In how far can existing techniques from knowledge engineering be used?

- The Maintenance Issue:
  How can one react to dynamic changes of the knowledge?

# Further Generalizations:

- Mix task 1 and task 2:
  Split dom(f) and find out which task to apply

- Mix task of type 2 with other tasks

Example:

Task Ind: Apply Inductive Reasoning

The INRECA-Approach:

Mixing Task of Type 2 and Task Ind

# Problem Solving Knowledge

In

- classical (procedural) programs
- knowledge based systems

the knowledge is used to solve a certain problem, e.g. to solve task 1.

(A) In a CBR-system the knowledge is used to solve tasks of type 2.

(B) If a system has some CBR-part, then the knowledge is in addition used to select the part of the knowledge used in the CBR-part

<u>Consequence:</u>
Methods for Knowledge Engineering should respect (A) and (B).

# Generalization:

Task of Type 2: For any a, b $\in$ dom(f)

decide the question
"Is the solution f(b) "good enough" to replace f(a)?"

"Good enough" has many interpretations, e.g.:

- f(b) is for further operations (almost) as good as f(a)
- f(a) can be easily determined from f(b) (adaption)

and others

The task of a CBR-system at compile time is essentially of type 2

Suppose $I = \{1,\ldots,n\}$; assume $J \subseteq I$:

$$X_J = \{x \in CB \mid x_i = a_i, i \in J\}, \quad X_i = X_{\{i\}}$$
$$m_J = \bigoplus (m_i \mid i \in J), \quad m_i = m_{\{i\}}$$

The sets $X_J$ are closed under intersections.

If $X_{J1} = X_{J2}$ for $J1 \neq J2$ we call it a multiplicity. Without multiplicities and conflicts, Dempster's rule simplifies and gives for $J' \subseteq J \subseteq I$

$$m_J(X_{J'}) = \prod_{i \in J'} g_i * \prod_{i \in J \setminus J'} (1-g_i)$$

$$= \sum_{J'' \subseteq J \setminus J'} \left( \prod_{i \in J'} g_i \right) * (-1)^{|J''|} * \prod_{k \in J''} g_k$$

Also:

$$m_J(CB) = \prod_{i \in J \setminus J'} (1-g_i) = 1 - \sum_{J'' \subseteq J \setminus J'} (-1)^{|J''|} * \prod_{k \in J''} g_k$$

Some $x \in CB$ may be elements of several focal sets X. Crucial assumption:
Each such membership contributes to the similarity of x and a according to the evidence measure of each X.

Definition:

(i) $\nu_J(X) = \sum_{Y \supseteq X} m_J(Y)$, Y a focal set for $m_J$

(ii) $\nu_J(x) = \nu_J(X)$, X the minimal focal set co uniquely defined).

(iii) $\mu^D_J(a,x) = \nu_J(x)$, where a is the actual case.

# Noise

$X_i^{e,d} = \{x \in CB \mid e \leq |X_i - a_i| \leq d\},$

$m_i^{e,d}(X_i^{e,d}) = g^{e,d},$

$m_i^{e,d}(CB) = 1 - \sum (g_i^{e,d} \mid (e,d))$

for $0 \leq e < d \leq 1$;

$g_i^{e,d}$ are again real numbers .

The rest is as above.

# Similarity and Utility

Plans, Configurations (sometimes Diagnoses) are not only

- Correct or incorrect

but also

- more or less useful

Hence we have two parameters

$\alpha$ : measures degree of correctness
$\beta$ : measures utility

Also, we have to consider
(Vocabulary, Similarity, Case Base)

plus
(Solution Transformation)

# Limitations of the Hamming Measure

$g = (g_1, ..., g_n)$ weight vector, $g_i \geq 0$

$$H_g(a,b) = \sum_{a_i \neq b_i} g_i \quad \text{weighted H - distance}$$

- The Hamming measure reflects importance
- The Hamming measure does not reflect dependencies

<u>Why $g_i \geq 0$ ?</u>

Otherwise there can be negative distances, e.g. $d(a, b) < 0 \leq d(a,a)$

<u>Hence:</u> No unrestricted use of negative weights

<u>Consequences:</u> Differences between attribute values cannot be expressed.

# One object - many cases

Often one connects

many problems with one object

i.e.

many cases with one object

Hence we need

all attributes for the problems considered

Each attribute needs

a justification

(for which problem is it useful?)

This allows the definition of a

case class

(all possible attributes)

Each case description is obtained from the case class by the

restriction to the justified attributes

# Objects versus Cases

- An object is defined by the primary attributes
- Each object gives rise to many problems

  an object may be

  - classified in various ways
  - planned
  - constructed

  ⋮

Each problem defines a case

Case description:

$$C = (A_1, \ldots, A_n,\ B_1, \ldots, B_m)$$

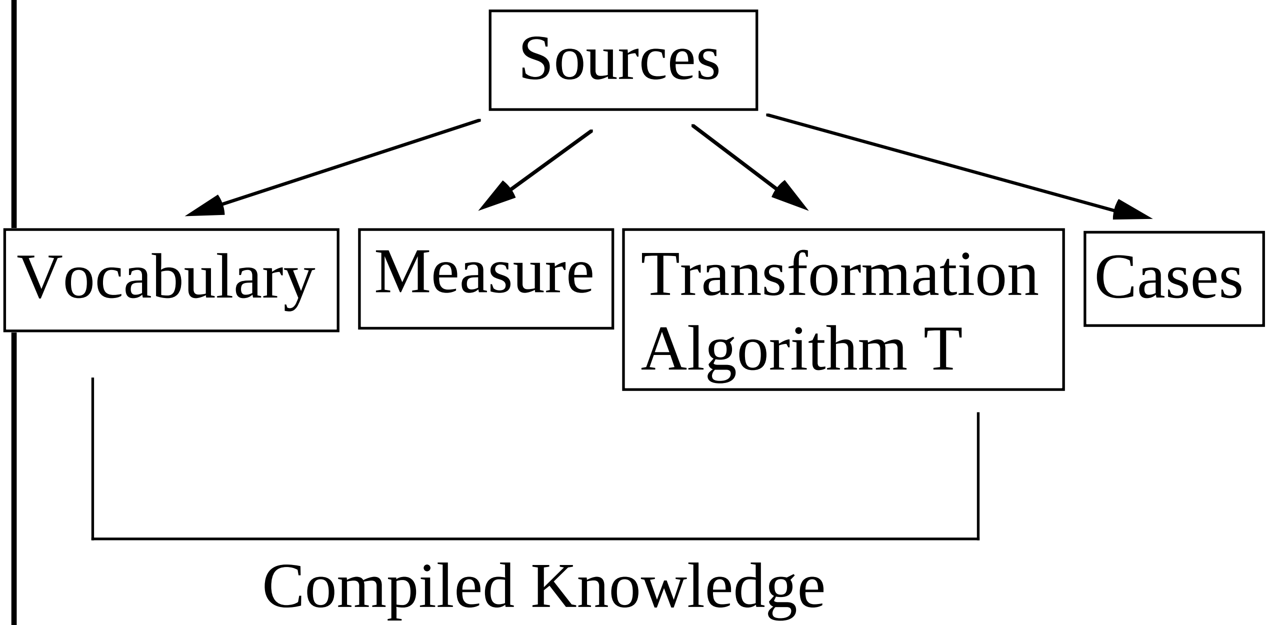
$A_i$: Selected primary attributes

$B_k$: Defined secondary attributes

The selection and definition of attributes is an important knowledge engineering task

# Solution Transformation

If solution transformations are present knowledge is distributed over items:




Extreme: All knowledge in T
(T is the problem solver)
Assumption: T always checks for correctness

# Semantics revisited

Similarity measure sim
and
solution transformation T
have to be considered as a unit.

Now: a actual case, $x \in CB$

<u>Utility :</u>

$\mu_{x,T} = f(\beta_1, \beta_2)$

where

$\beta_1$ measures cost of applying T to the solution of x

$\beta_2$ measures degree of optimality of the solution

# How to find secondary attributes

This is a knowledge acquisition task.

<u>Assumption:</u> The expert can (intuitively) decide S (z, x, y)

<u>Scenario:</u>
- Present z, y to the expert
- Select i such that $z_i \neq y_i$
- Obtain x from y by changing $y_i$ to $z_i$
- Ask the expert: S(z, x, y) ?

<u>If yes:</u> Indication for attribute i independent from the rest of attributes

<u>If no:</u> Ask the expert: Why?

<u>If the answer:</u> "You have to change some $y_J$ too," then two dependent attributes $A_i$ and $A_j$ are found.

Figure out the dependency f (i, j) and create a new attribute

# The XOR Example

$U = \{(0,0), (0,1), (1,0), (1,1)\}$
$K_1 = \{(0,0), (1,1)\}$, $K_2 = U \setminus K_1$

Observation:
If $CB \subseteq U$, $|CB| = 2$
then for no weighted Hamming measure Hg
(C B, Hg) can classify $(K_1, K_2)$ correctly
using NNP.

Two possibilities:

1. Use other measures which can carry more knowledge
2. Use a new secondary attibute $x_3$, $x_3 = x_1 \oplus x_2$

Example:

$$f(x_1,..,x_n,y_1,...y_n) = \begin{cases} 1 & \text{if } x_i = y_i \text{ all } i \\ 0 & \text{else} \end{cases}$$

$$X = \{x_1,..,x_n\} \quad , Y = \{y_1,..,y_n\}$$

$$H_f(X) = H_f(Y) = 0 , \; H_f(X,Y) \approx \frac{n}{2^n}$$

$$\underline{H}_f(X) = \underline{H}_f(Y) = n$$

$$\underline{H}_f(X,Y) = 1$$

$$I_1(X,Y) \approx - \frac{n}{2^n} \approx 0$$

$$I_2(X,Y) = 2n - 1$$

# The Influence Measure

<u>Def</u>: The influence measure is the generalised Hamming measure given by the weights $g_J = \inf_f (J)$

<u>Observations:</u>

- $g_I$ = number of classes
- there may be $J \subseteq I$ with $g_J > g_I$ ($\inf_f$ is not monotonic)
- f is difficult to compute

<u>Task:</u> Determine those J which

- are small
- have large influence

# Influence versus Entropy

$\underline{H}_f\,(J) = \log\,(\,\inf_f\,(J))$

behaves like an entropy potential

$I_2\,(J,J') = \underline{H}_f\,(J) + \underline{H}_f\,(J') - \underline{H}_f\,(J \cup J')$

$H_f\,(J)$ measures importance of J to y

$\underline{H}_f\,(J)$ measures importance of J to y and I \ J

# Entropy Potential

$f(x_1,..,x_n) \longrightarrow y$

Consider $x_1,.., x_n, y$ as random variables

For $J \subseteq \{x_1,..,x_n,y\}$ : $H(J)$ entropy

Cross - Entropy:

$$H_f(J) = H(J) + H(y) - H(J \cup \{y\})$$

Dependencies:

$$I_1(J, J') = H_f(J) + H_f(J') - H_f(J \cup J')$$

# Semantics of Similarity

The meaning of the relations should be

For any z the choice of x
such that NN( z, x)
is the "best possible"

This is NNP : Nearest - Neighbor - Principle

How can it be justified?

If the relations are obtained from a measure sim,
what is the meaning of the numerical values of sim?

# Evidences

Suppose we know the value $a_i$ of the actual case a.
This is a piece of information!

It gives some evidence that the
NN of a is in

$$X_i = \{ x \in CB \mid a_i = x_i \}$$

If no other information is present, elements of $X_i$ are not distinguished.

The evidence

- may objective ( model based) or subjective
- comes from expert knowledge
- may be very small

# Evidences

<u>weight of the evidence:</u>

$$m_i ( X_i) = g_i$$

<u>Ignorance:</u>

$$m_i ( CB) = 1 – g_i$$

$$m_i ( Y) = 0 \text{ for all other } Y \subseteq CB$$

$m_i$ is a Dempster - measure on $\wp$(CB)

Two measures $m_i$ and $m_j$ can be accumulated to $m_i \oplus m_j$ .
Dempster's rule computes this for independent observations.

# Summary

Semantics:

- Correctness: Leads to the notion of approximate truth. One approach is according to evidence theory
- Optimality: Leads to preferences and utility
- A formal semantics should incorporate both.

# Summary

Knowledge Engineering:

- The knowledge sources should be investigated:
  - Are there clearly described cases?
  - Are the primary attributes collected?
  - What kind of background knowledge is present and useful?
- How is the knowledge best distributed over (attributes, measure, case base, solution transformation) ?
  This is a pragmatic decision!
- Knowledge acquisition and information retrieval techniques should adapted to distribute knowledge
- Learning techniques should be applied

# Summary

Maintenance:

- Compiled knowledge:
    - Updating is difficult as in knowledge based systems
    - If learning has been applied it could be continued

- Interpreted knowledge:
    - Updating is easier; it results in the updating of the case base

Moral: Compile

- as little knowledge as possible
- as much knowledge as absolutely necessary.

# CBR

CBR has many

- applications
- aspects

- Classification, Diagnosis

- Configuration

- Planning

- Decision Support

.
.
.

# Compilation versus Interpretation

Compilation process comp ("Coding")
Interpretation process Int

Compilation $\xrightarrow{\text{yes}}$ costs at compile time
Compilation $\xrightarrow{\text{less}}$ costs at run time
Interpretation $\xrightarrow{\text{none}}$ costs at compile time
Interpretation $\xrightarrow{\text{all}}$ costs at run time

More Knowledge in Sim

$\Rrightarrow$

better classification, smaller CB,
but application of Sim possibly more expensive

<u>Simple Cost Function:</u>

Costs = C + n P
C = Compilation Costs
P = Cost for one Solution
n = Number of Applications

# Attributes

Two Sorts of attributes:

1 Primary attributes: Values come from the available information sources.

2 Secondary attributes: Are defined in terms of primary attributes.

- Primary attributes contain domain Knowledge

- Secondary attributes contain task knowledge

Example: Customers of a bank

Primary attributes:

$A_1$ : Income
$A_2$ : Spending
$A_3$ : Interest rate on savings account

Secondary attributes:

$A_4$ : $A_1 - A_2$
$A_5$ : (maximal interest rate available today) $- A_3$

Classification tasks:

1) Good customers : $A_4 \geq 0$

2) Customers that may change their bank : $A_5 > 0$

# Dependencies

Attributes $A_i$, $i \in F$;

Classification $f: U \to \{ 1, .., n \}$

k-ary dependencies between attributes
$\sim$
subsets $J \subseteq I$, $|J| = k$

Def: Generalized Hamming Distance :

weights $g_J$ for each $J \subseteq I$

$GH(a,b) = \sum (g_J \mid J \subseteq I, a \mid J \neq b \mid J)$

Specializations for 2-ary, 3-ary,... dependencies.

Question: How to choose the $g_J$ ?
This means: Which $J \subseteq I$ are important?
This is a - priori - knowledge, to be compiled

Again: - objective approach (model-based)
- subjective approach

# The Influence Potential

Notation: $U_J$ : Restriction to Attributes $A_i$, $i \in J$

Def: (i) $a_J \equiv_f b_J$ for $a_J, b_J \in U_J$

$\Leftrightarrow$

for all $c \in U_{I \setminus J}$ : $f(a_J, c) = f(b_J, c)$

(ii) The influence of J is

$\inf_f(J) := | \; U_J / \equiv_f |$

The influence of $J \subseteq I$ is the number of different restrictions to $I \setminus J$ of the classifying function f.

## Observations:

- the influence potential reflects dependencies
- the influence potential is in general not known
- estimates are often subjective and reflect expert knowledge
- the Hamming distance corresponds to singletons $\{i\} \subseteq I$.
- one can approximate GH by knowing or estimating $\inf_f(J)$ for $|J| = 2,3,\ldots$
- to estimate $\inf_f(J)$ is often easier than to know the exact dependencies

# A suggestion for Semantics (sim,T)

Actual case: a

Observed attributes: indexed by J

Minimal focal set: $X \subseteq CB$

Accumulated evidence: $\nu_J(X)$

Simplifying assumption: All cases in CB have optimal solutions

Reasonable definition for $x \in X$:
$\mu_J(a,x) := \nu_J(x) \cdot \mu_{x,T}(a)$

This grasps

- degree of correctness
- utility