Lecture 2
Case-based classification

Thierry Denœux

tdenoeux@utc.fr

University of Compiègne, France

Objective

- Apply to TBM to pattern classication
- Classification (discrimination) = assignment of objects to predefined categories (classes)
- Applications:
  - character, speech recognition
  - diagnosis, condition monitoring
  - target discrimination, face recognition, person identification
  - text categorization, context-based image retrieval, web mining, etc.

The approach

- distance/case/instance/memory-based learning
- Basic principle: assess the similarity between the current pattern to be classified and each of n patterns in a data base (learning set)
- A very general paradigm:
  - k-nearest neighbor, kernel classifiers
  - radial basis function, Learning vector quantization neural networks
  - fuzzy system classifiers

Why use the TBM ? (1/2)

1. Classifier ouput = belief function
   - more faithfull description of uncertainty, distinct representation of
     - ignorance (pattern dissimilar from all training examples)
     - conflicting information (pattern similar to examples of different classes)
   - greater robustness and improved performance when combining several classifiers (e.g. sensor fusion)

Why use the TBM ? (2/2)

2. Possibility to handle weak learning information:

   - Partial knowledge of the class of learning examples (e.g., $o \in \{\omega_1, \omega_2\}$, $o \notin \omega_3$, $P(o \in \omega_1) = 0.7$, etc.)
   - heterogeneous, non exhaustive learning sets:
     - DB_1 with objects from $\{\omega_1, \omega_2\}$ and attributes x_j, $j \in J$
     - DB_2 with objects from $\{\omega_2, \omega_3\}$ and attributes x_j, $j \in J' \neq J$

Notations/Formalization

- Population $\mathcal{P}$ of objects, each object o described by:
  - $\mathbf{x}$: vector of d attributes (features), quantitative, qualitative, mixed
  - c: class/category/group, qualitative, values in finite set $\Omega = \{\omega_k\}_{k=1}^{K}$.
- For an object o, $\mathbf{x}$ is observed, c is not
- based on available learning information $\mathcal{L}$, we want to assess our beliefs regarding the value of c in the form of belief function $bel_c^{\Omega}[\mathbf{x}, \mathcal{L}]$

Learning information

- Learning/training set

$$\boxed{\mathcal{L} = \{e_i\}_{i=1}^{n}}$$

  of learning examples (training patterns) e_i with completly or partially known classification.

- Simplest form ("classical learning set"): $e_i = (\mathbf{x}_i, c_i)$
- More general form: $e_i = (\mathbf{x}_i, A_i)$ where $A_i \subseteq \Omega$ is a set possible values for c_i (Remark: generalizes both unsupervised and supervised learning).

Case-based inference

- The problem: Given a learning set

$$\mathcal{L} = \{e_i = (\mathbf{x}_i, A_i)\}_{i=1}^{n} \quad (A_i \subseteq \Omega)$$

  compute $bel_c^{\Omega}[\mathbf{x}, \mathcal{L}]$ (Your belief concerning the class c of a new object described by feature vector $\mathbf{x}$).

- Fundamental principle (FP): Two objects are all the more likely to belong to the same class that their feature vectors are more similar.

Formalization

- Let o and o' be 2 objects, with feature vectors $\mathbf{x}$ and $\mathbf{x}'$, and classes c and c', resp.
- The proposition 'o and o' belong to the same class' corresponds to $S = \{(\omega_k, \omega_k)\}_{k=1}^{K} \subseteq \Omega^2$.
- The fundamental principle can be expressed as:

$$m^{\Omega^2}_{(c,c')}[\mathbf{x}, \mathbf{x}'](S) = \alpha(\mathbf{x}, \mathbf{x}')$$

$$m^{\Omega^2}_{(c,c')}[\mathbf{x}, \mathbf{x}'](\Omega^2) = 1 - \alpha(\mathbf{x}, \mathbf{x}')$$

where $\alpha(\cdot, \cdot)$ is a similarity measure taking values in $[0, 1]$ ($\alpha(\mathbf{x}, \mathbf{x}) \leq \alpha(\mathbf{x}, \mathbf{x}')$ for all $\mathbf{x} \neq \mathbf{x}'$).

Impact of 1 example (1)

- Let $e_i = (\mathbf{x}_i, A_i)$ be a learning example. Our belief concerning c_i is represented by a bba $m^{\Omega}_{c_i}$ such that $m^{\Omega}_{c_i}(A_i)$=1.
- Let $\mathbf{x}$ be the feature vector for a new object o to be classified. From the FP, we have a bba $m^{\Omega^2}_{(c,c_i)}[\mathbf{x}, \mathbf{x}_i]$.
- The bba describing our knowledge of the class of o can be obtained as:

$$m^{\Omega}_{c}[\mathbf{x}, e_i] = \left((m^{\Omega}_{c_i})^{\uparrow\Omega^2} \textcircled{\cap} m^{\Omega^2}_{(c,c_i)}[\mathbf{x}, \mathbf{x}_i] \right)^{\downarrow\Omega}$$

Impact of 1 example (2)

- Corresponding bba:

$$m_c^{\Omega}[\mathbf{x}, e_i](A) = \begin{cases} \alpha(\mathbf{x}, \mathbf{x}_i) & \text{if } A = A_i \\ 1 - \alpha(\mathbf{x}, \mathbf{x}_i) & \text{if } A = \Omega \\ 0 & \text{otherwise} \end{cases}$$

- $m_c^{\Omega}[\mathbf{x}, e_i]$ is
  - maximally specific when $\mathbf{x} = \mathbf{x}_i$
  - vacuous when $\mathbf{x}$ and $\mathbf{x}_i$ are maximally dissimilar ($\alpha(\mathbf{x}, \mathbf{x}_i) = 0$).

Definition of $\alpha(\cdot, \cdot)$

- A general form: $\alpha(\mathbf{x}, \mathbf{x}_i) = \phi(\delta(\mathbf{x}, \mathbf{x}_i))$, with δ a distance (or a dissimilarity) measure and ϕ decreasing function such that $0 < \phi(0) \leq 1$ and $\lim_{\delta \to \infty} = 0$.
- A natural choice when $\mathbf{x}, \mathbf{x}_i \in \mathbb{R}^d$: the Euclidean distance

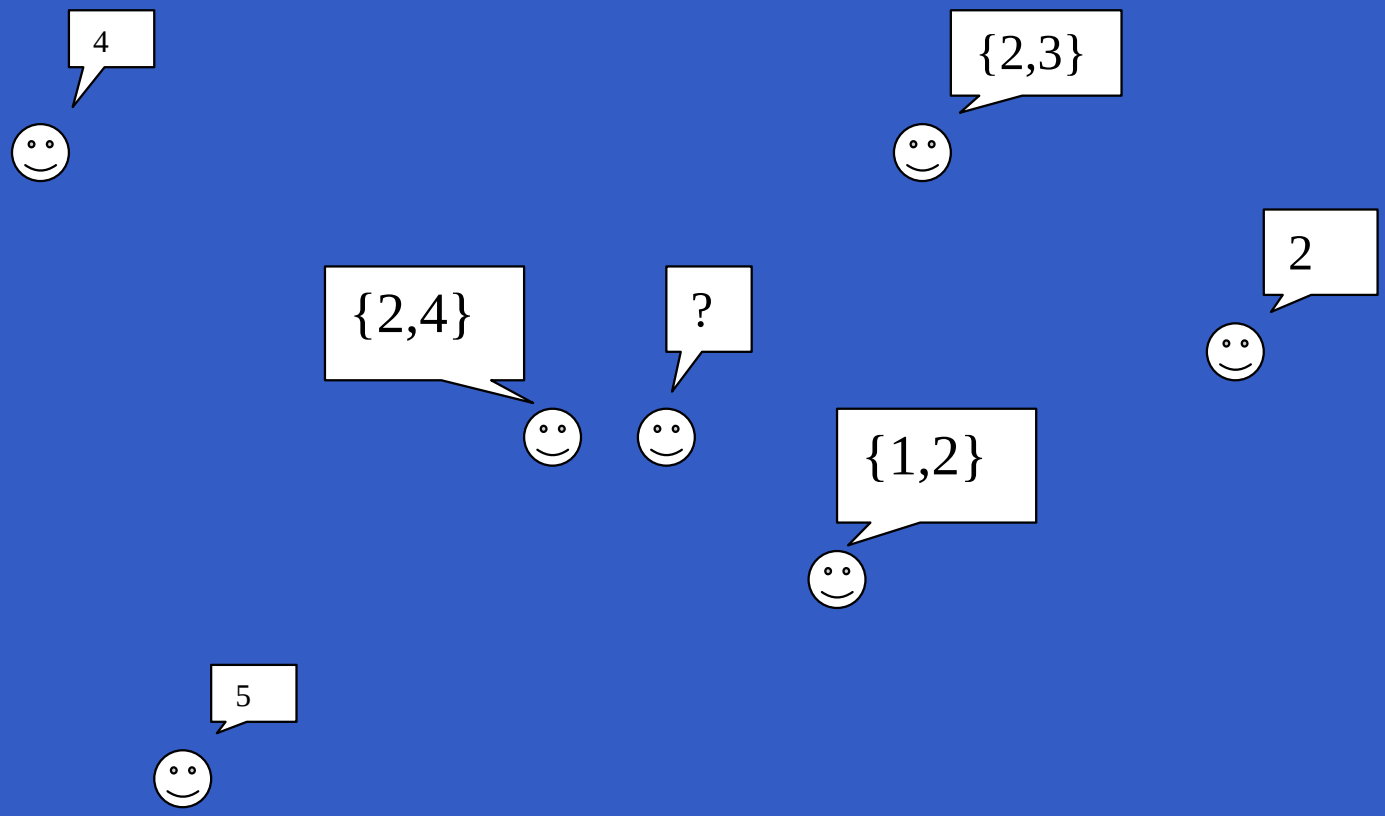
$$\alpha(\mathbf{x}, \mathbf{x}_i) = \alpha_0 \exp\left(-\gamma \sum_{j=1}^{d} (x_j - x_{ij})^2\right)$$

with $0 < \alpha_0 \leq 1$ and $\gamma > 0$.

Impact of n examples

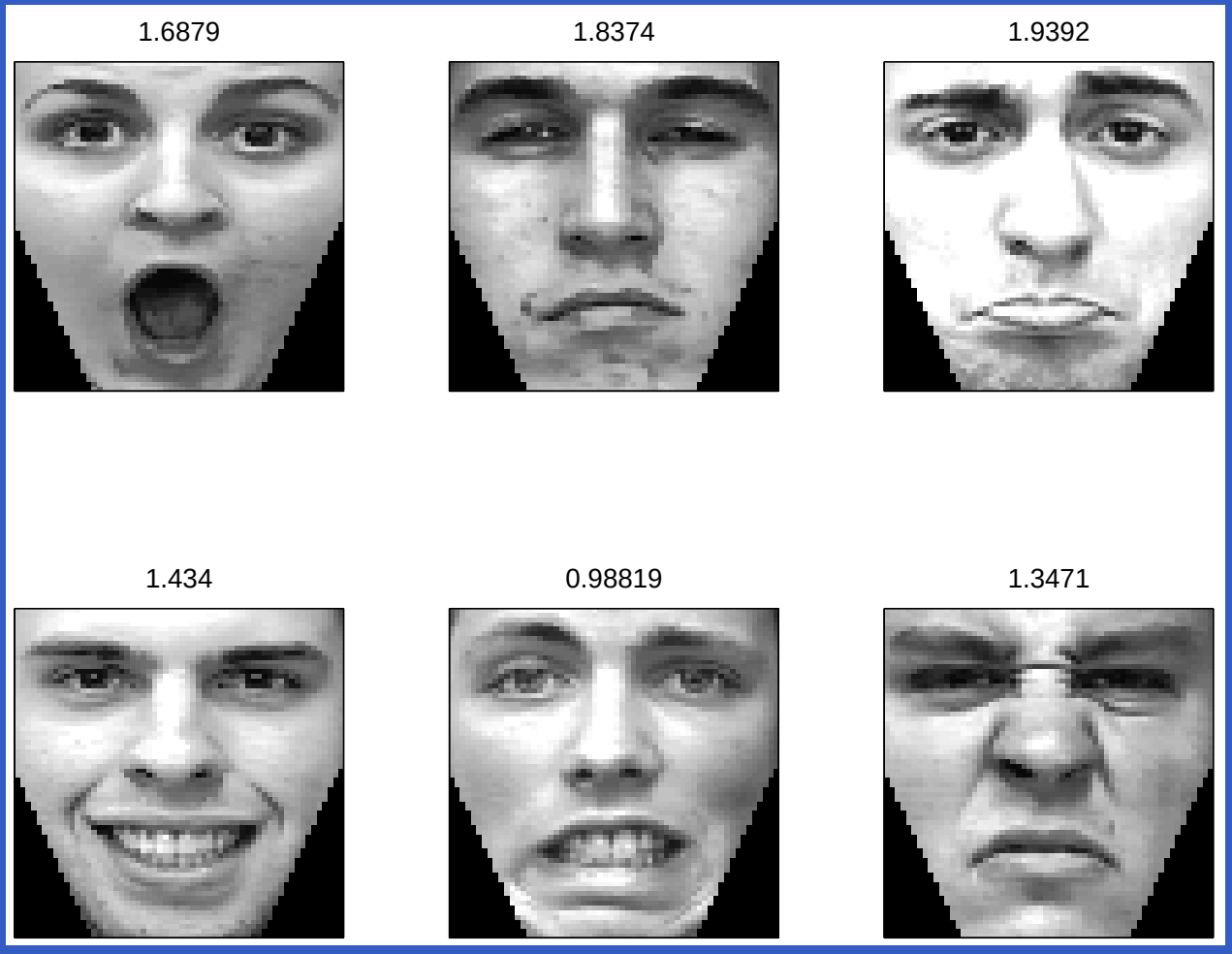
$$\left.\begin{array}{c} e_1 \rightarrow m_c^{\Omega}[\mathbf{x}, e_1] \\ \vdots \\ e_i \rightarrow m_c^{\Omega}[\mathbf{x}, e_i] \\ \vdots \\ e_n \rightarrow m_c^{\Omega}[\mathbf{x}, e_n] \end{array}\right\} \longrightarrow \boxed{m_c^{\Omega}[\mathbf{x}, \mathcal{L}] = \textcircled{\cap}_{i=1}^{n} m_c^{\Omega}[\mathbf{x}, e_i]}$$

An anthropomorphic model ...



Example: Facial expression recognition

Example: Facial expression recognition



Example: calculations

	$m[e_1]$	$m[e_2]$	$m[e_3]$	$m[e_4]$	$m[e_5]$	$m[e_6]$	$m[\{e_i\}_1^6]$
Su	0.17	0	0	0	0	0	0.08
An	0	0.14	0	0	0	0	0.06
Sa	0	0	0.13	0	0	0	0.06
Jo	0	0	0	0.21	0	0	0.11
Fe	0	0	0	0	0.34	0	0.19
Di	0	0	0	0	0	0.23	0.12
Ω	0.83	0.86	0.87	0.79	0.67	0.77	0.38

Evidential k-NN rule

- Let $\mathbf{x}_{(1)}, \ldots, \mathbf{x}_{(n)}$ denote the training vectors arranged in the order of increasing distance to $\mathbf{x}$.
- For k large enough,

$$m^{\Omega}[\mathbf{x}, e_{(\ell)}](\Omega) \approx 1 \quad \forall \ell > k$$

so that

$$\boxed{m[\mathbf{x}, \mathcal{L}] \approx \textcircled{\cap}_{i=1}^{k} m[\mathbf{x}, e_{(i)}]}$$

- Efficient algorithms allow to find the k-NN to $\mathbf{x}$ without calculating all the n distances.

Advanced issues

1. More general training data
2. Decision making
3. Learning

Learning information: general case

Most general situation: $\boxed{e_i = (\mathbf{x}_i, m_i^\Omega)}$

- m_i^Ω: a bba representing Your partial knowledge regarding the class of object i.
- Special cases:
  - $m_i^\Omega(\{\omega_k\}) = 1$: precise (standard) labeling
  - $m_i^\Omega(A) = 1$ for $A \subseteq \Omega$: imprecise labeling
  - m_i^Ω is a proba. function: probabilistic labeling
  - m_i^Ω has nested focal elements: possibilistic labeling ("object i is big"), etc...

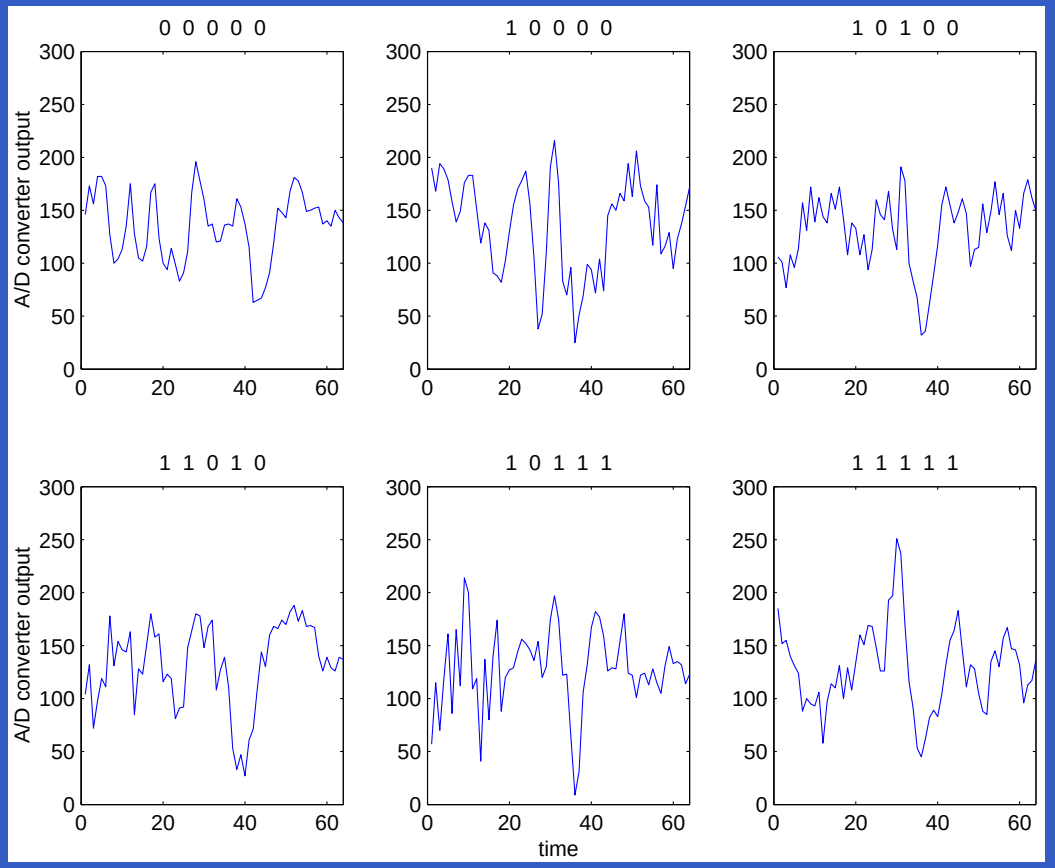
Learning information: general case

- Special case $e_i = (\mathbf{x}_i, A_i)$: $m_c^\Omega[\mathbf{x}, e_i]$ is a discounting of m_i with $m_i(A_i) = 1$ and discounting factor $1 - \alpha(\mathbf{x}, \mathbf{x}_i)$.
- General case $e_i = (\mathbf{x}_i, m_i)$ with m_i arbitrary bba

$$m_c^\Omega[\mathbf{x}, e_i](A) = \begin{cases} \alpha(\mathbf{x}, \mathbf{x}_i) m_i(A) & \forall A \neq \Omega \\ 1 - \alpha(\mathbf{x}, \mathbf{x}_i)(1 - m_i(\Omega)) \text{ if } A = \Omega \end{cases}$$

Example: EEG data (1)

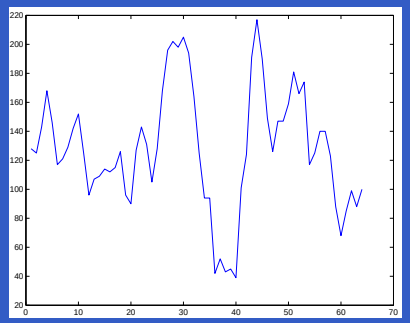
500 EEG signals encoded as 64-D patterns, 50 % pos. (K-complexes), 50 % neg. (delta waves), 5 experts.



Example: EEG data (2)

$$\Omega = \{K\text{-complex}, \delta\text{-wave}\}$$

$\mathbf{x}_i$=

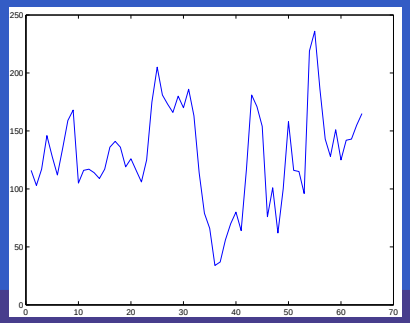


A	$\emptyset$	$\{K\}$	$\{\delta\}$	Ω
$m_i(A)$	0	0.8	0.2	0

$\updownarrow \alpha(\mathbf{x}, \mathbf{x}_i) = 0.5$

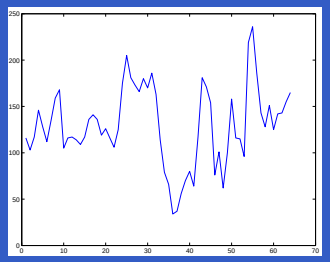
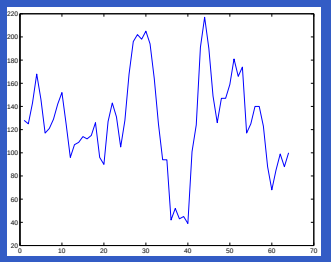
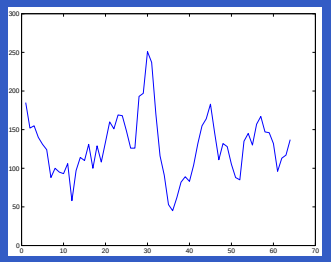
$\downarrow$ discounting

$\mathbf{x}$=



A	$\emptyset$	$\{K\}$	$\{\delta\}$	Ω
$m[\mathbf{x}, e_i](A)$	0	0.4	0.1	0.5

Example: EEG data (3)

$\mathbf{x}=$  $\mathbf{x}_i=$  $\mathbf{x}_j=$ 

$\alpha(\mathbf{x}, \mathbf{x}_i) = 0.5 \quad \alpha(\mathbf{x}, \mathbf{x}_j) = 0.7$

A	$\emptyset$	$\{K\}$	$\{\delta\}$	Ω
$m_i(A)$	0	0.8	0.2	0
$m_j(A)$	0	0.6	0.4	0
$m[\mathbf{x}, e_i](A)$	0	0.4	0.1	0.5
$m[\mathbf{x}, e_j](A)$	0	0.42	0.28	0.3
$m[\mathbf{x}, e_i, e_j](A)$	0.196	0.282	0.372	0.15

Decision analysis (1)

- Pb: given $bel_c^{\Omega}[\mathbf{x}, \mathcal{L}]$, how to make a decision ?
- General approach: compute the pignistic probability distribution $BetP[\mathbf{x}, \mathcal{L}]$, and apply the classical Bayesian decision analysis.
- We need to define:
  - the set of actions: $\mathcal{A} = \{a_1, \ldots, a_r\}$
  - the loss function $\lambda(a, \omega)$, $\forall a \in \mathcal{A}, \forall \omega \in \Omega$
- Choose a that minimizes the pignistic risk:

$$R(a) = \sum_{\omega \in \Omega} \lambda(a, \omega) BetP[\mathbf{x}, \mathcal{L}](\omega)$$

Decision analysis (2)

- A critical issue: is the learning set exhaustive or not ?
- Case 1: all possible classes are represented in the training set. Let
  - $\Omega = \{\omega_1, \ldots, \omega_K\}$ = set of classes
  - $\mathcal{A} = \{a_0, a_1, \ldots, a_K\}$ = set of actions with a_k= assignment to class ω_k, and a_0=rejection.

$$\lambda(a_k, \omega_\ell) = \begin{cases} 0 & k = \ell, \quad k, \ell \in \{1, \ldots, K\} \\ 1 & k \neq \ell, \quad k, \ell \in \{1, \ldots, K\} \\ \lambda_0 & k = 0, \quad \ell \in \{1, \ldots, K\} \end{cases}$$

Decision analysis (3)

Then we have

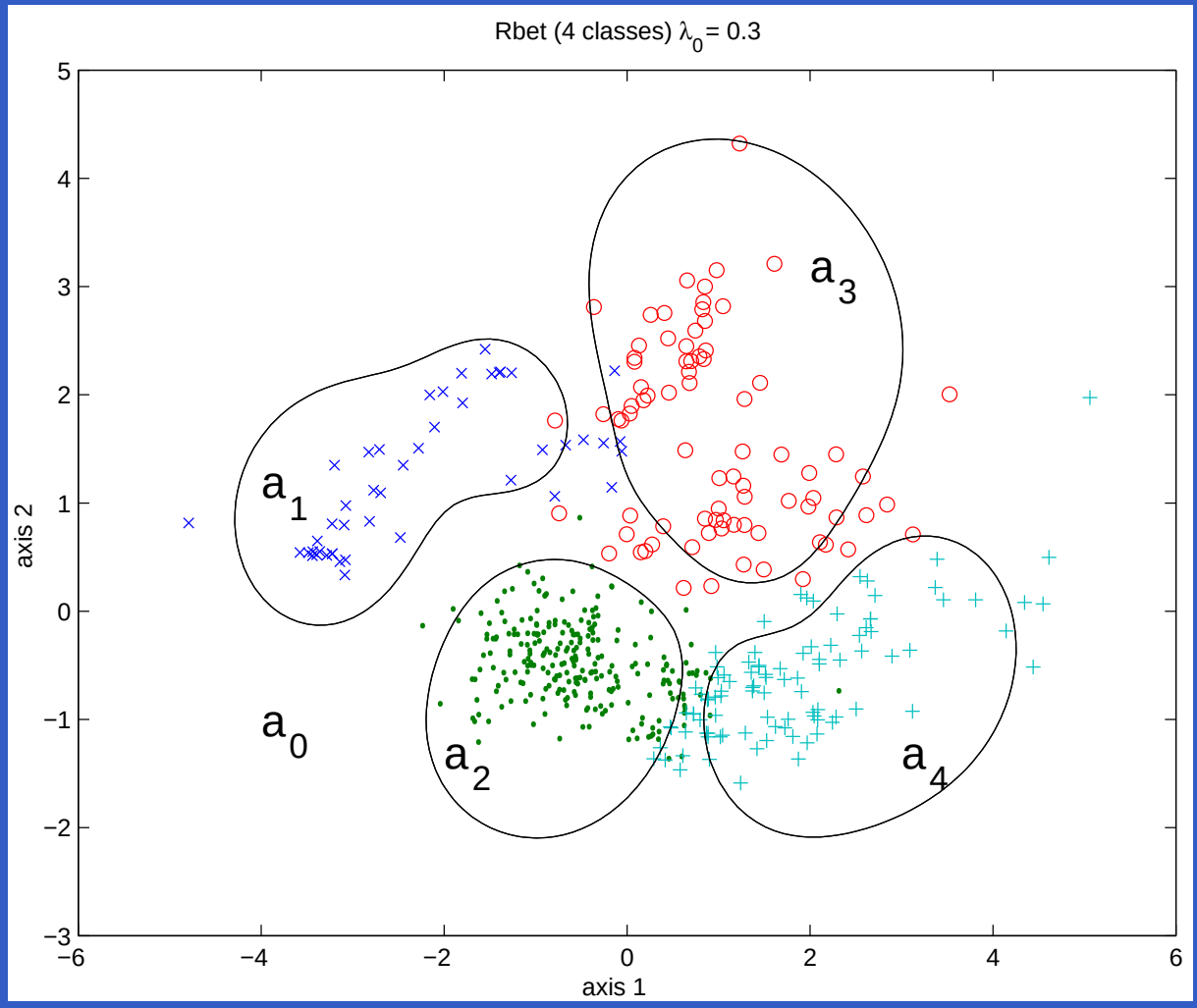
$$R(a_k) = 1 - BetP[\mathbf{x}, \mathcal{L}](\omega_k)$$

$$R(a_0) = \lambda_0$$

So the decision rule is

$$D(\mathbf{x}) = \begin{cases} a_k & \text{if } BetP(\omega_k) > BetP(\omega_\ell) \quad \forall \ell \neq k \\ & \text{and } BetP(\omega_k) > 1 - \lambda_0 \\ a_0 & \text{otherwise} \end{cases}$$

Decision analysis (4)



Decision analysis (5)

Case 2: There is (may be) an unknown class ω_u not represented in $\mathcal{L}$. Then $\Omega = \{\omega_1, \ldots, \omega_K, \omega_u\}$ and

$$\mathcal{A} = \{a_0, a_1, \ldots, a_K, a_u\}$$

There is no evidence in the learning set that points to ω_u, so $bel^{\Omega}[\mathbf{x}, \mathcal{L}](\{\omega_u\}) = 0$. However

$$BetP(\omega_u) = \frac{m[\mathbf{x}, \mathcal{L}](\Omega)}{K+1}$$

For certain loss functions, assignment to the unknown class is possible !

Decision analysis (6)

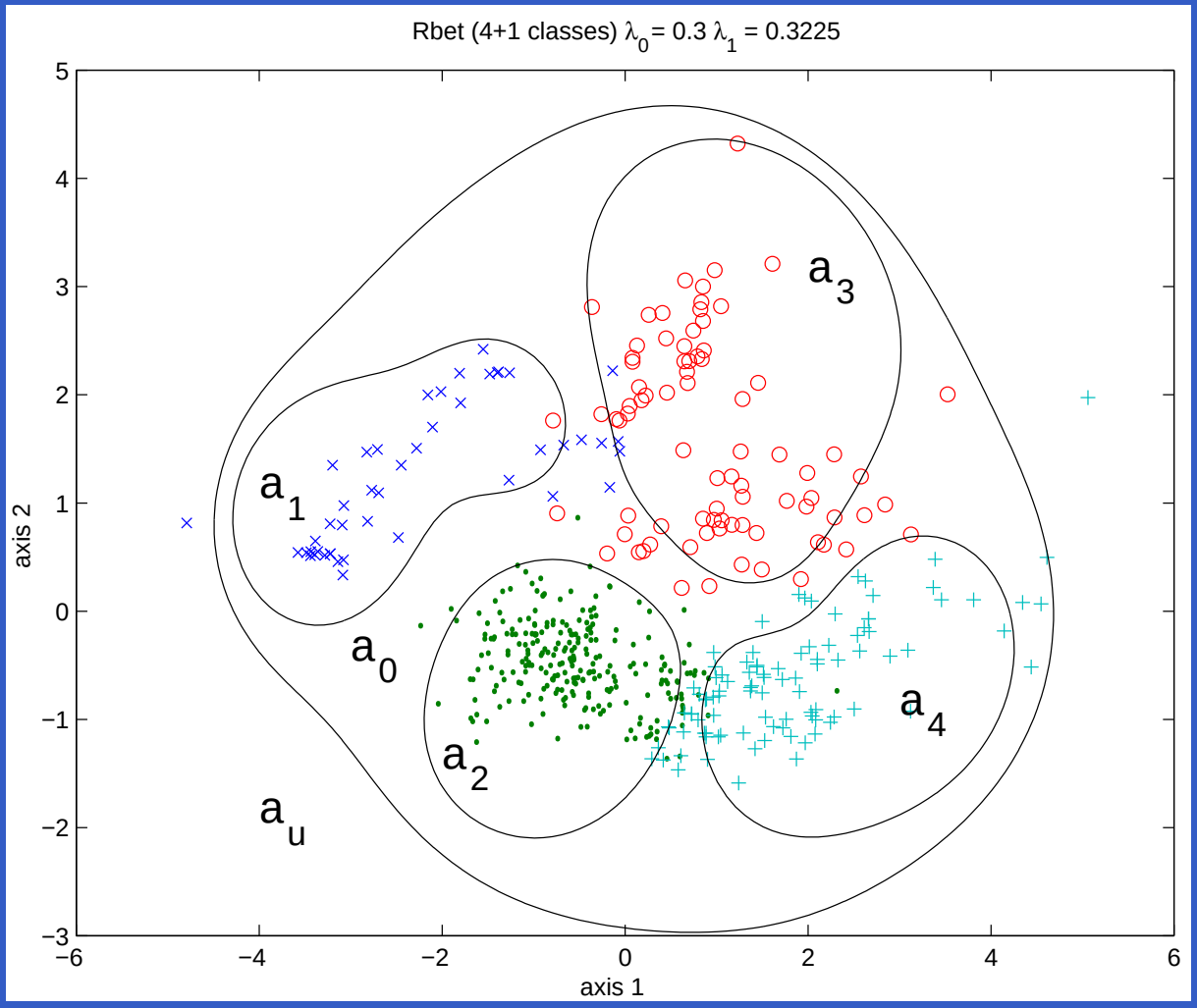
Assume

	a_0	a_1	a_2	$\dots$	a_{K-1}	a_K	α_u
ω_1	λ_0	0	1	$\dots$	1	1	λ_1
ω_2	λ_0	1	0	$\dots$	1	1	λ_1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
ω_{K-1}	λ_0	1	1	$\dots$	0	1	λ_1
ω_K	λ_0	1	1	$\dots$	1	0	λ_1
ω_u	λ_0	1	1	$\dots$	1	1	0

Then,

$$R(a_u) = \lambda_1 \left(1 - \frac{m[\mathbf{x}, \mathcal{L}](\Omega)}{K+1} \right)$$

Decision analysis (7)



Learning (1)

- Basic model:

$$\alpha(\mathbf{x}, \mathbf{x}_i) = \alpha_0 \exp\left(-\gamma \sum_{j=1}^{d} (x_j - x_{ij})^2\right)$$

- Pb: determination of α_0 and γ.
- Simple approach: fix to "reasonable values"

$$\alpha_0 = 0.9 \qquad \gamma = 1/\overline{\delta}_k$$

with $\overline{\delta}_k$ mean squared Euclidean distance between a learning vector and one of its k NN's.

Learning (2)

- More powerful approach: minimize an empirical error criterion using the leave-one-out method.
- Principle:
  - classify each learning example $\mathbf{x}_i$ using the other training patterns,
  - compare the result bba $m_{c_i}[\mathbf{x}_i, \mathcal{L}^{-i}]$ with the class label m_i, compute error $E_i(\alpha_0, \gamma)$
  - Minimize $E(\alpha_0, \gamma) = \sum_{i=1}^{n} E_i(\alpha_0, \gamma)$ using a gradient-based optimization procedure.

Learning (2)

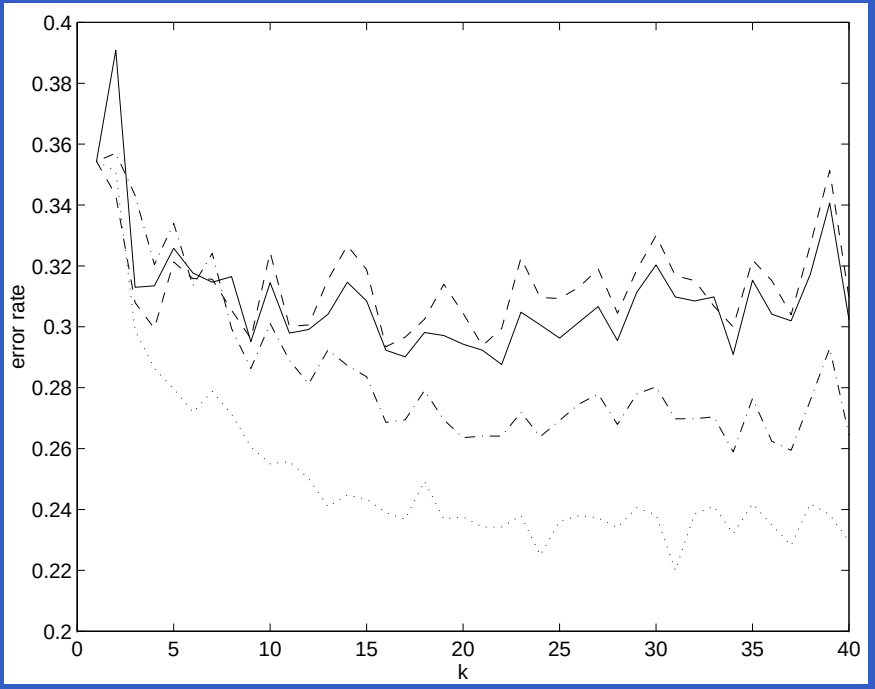
- Pb: which error function to compare 2 belief functions ?
- One solution:

$$E_i = 1 - \sum_{k=1}^{K} BetP_i(\omega_k) BetP_{c_i}[\mathbf{x}_i, \mathcal{L}^{-i}](\omega_k)$$

- Property: m_i vacuous $\Rightarrow E_i = 1 - 1/K$ whatever α_0 and γ (if nothing is known concerning the class of example i, that example has no influence on the result).

Learning (3): example of results

Gaussian data, $n = 300$, $K = 3$, $d = 10$. Comparison with different k-NN rules: voting (-), fuzzy (- -), distance-weighted (-.).



Learning: more complex models (1)

- The performances of the method depends on the particular distance measure used.
- One solution: optimize the distance measure

$$\alpha(\mathbf{x}, \mathbf{x}_i) = \alpha_0 \exp\left(-\gamma \sum_{j=1}^{d} w_j (x_j - x_{ij})^2\right)$$

- Error criterion:

$$J(\alpha_0, \gamma, \mathbf{w}) = E(\alpha_0, \gamma, \mathbf{w}) + \mu \sum_{j=1}^{d} (w_j - 1)^2$$

Example: expression recognition

Surprise/Disgust

erreur rate = 4 %

Joy/sadness

erreur rate = 0.8 %

(35 examples in each class)

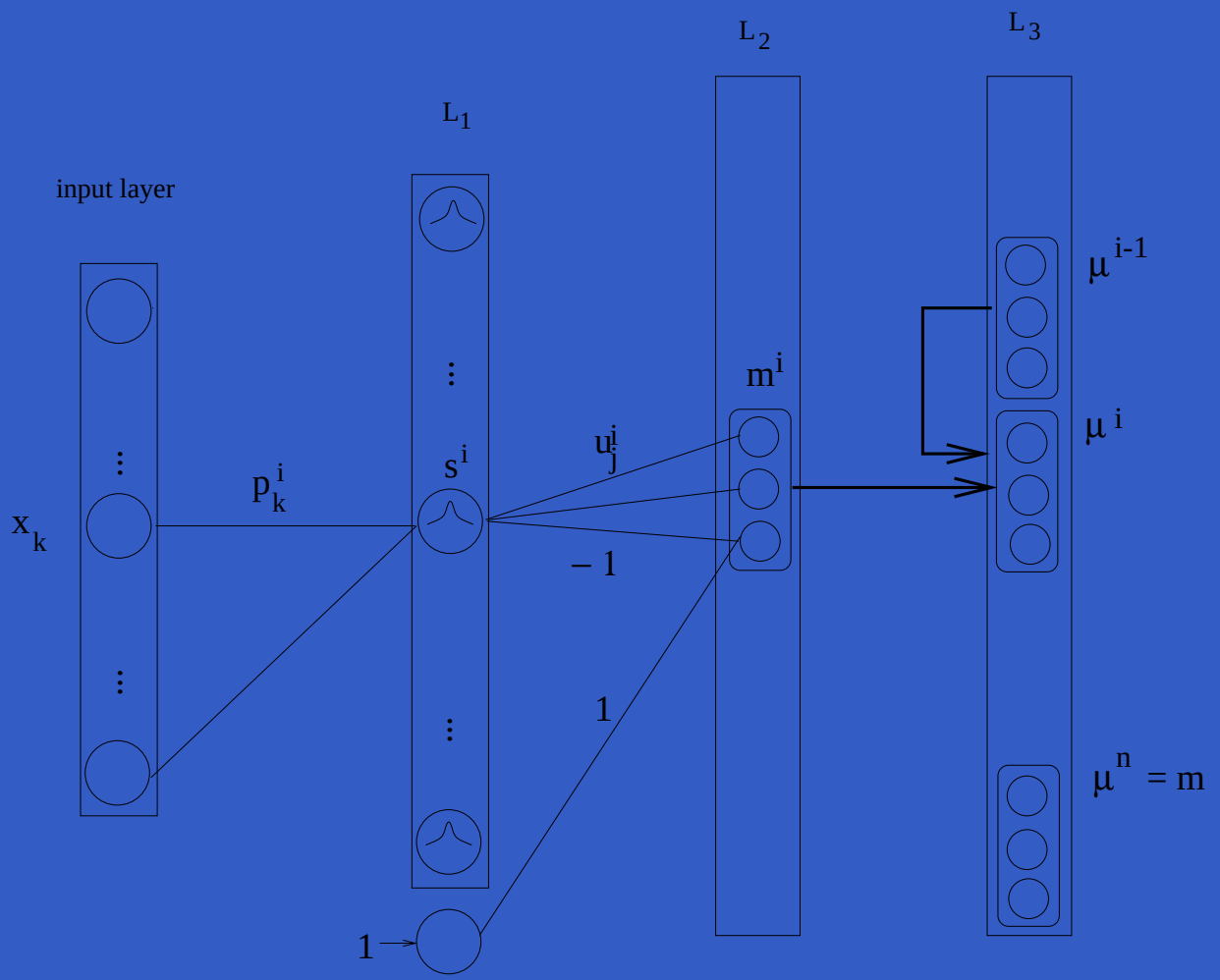
Classification using prototypes

- Idea: to speed up calculations, summarize the learning set as r reference patterns (prototypes): $\mathbf{p}_1, \cdots, \mathbf{p}_r$.
- Each prototype i has degree of membership u_{ik} to each class ω_k with $\sum_{k=1}^{K} u_{ik} = 1$
- The similarity to each prototype induces a bba:

$$\begin{aligned} m[\mathbf{x}, \mathbf{p}_i](\{\omega_k\}) &= \alpha_i u_{ik} \exp(-\gamma_i \|\mathbf{x} - \mathbf{p}_i\|^2) \quad \forall k \\ m[\mathbf{x}, \mathbf{p}_i](\Omega) &= 1 - \alpha_i \exp(-\gamma_i \|\mathbf{x} - \mathbf{p}_i\|^2) \end{aligned}$$

$$m[\mathbf{x}, \mathcal{L}] = \textcircled{\cap}_{i=1}^{r} m[\mathbf{x}, \mathbf{p}_i]$$

Neural network implementation



Results on 'classical data'

Vowel data

$K = 11$,

$d = 10$

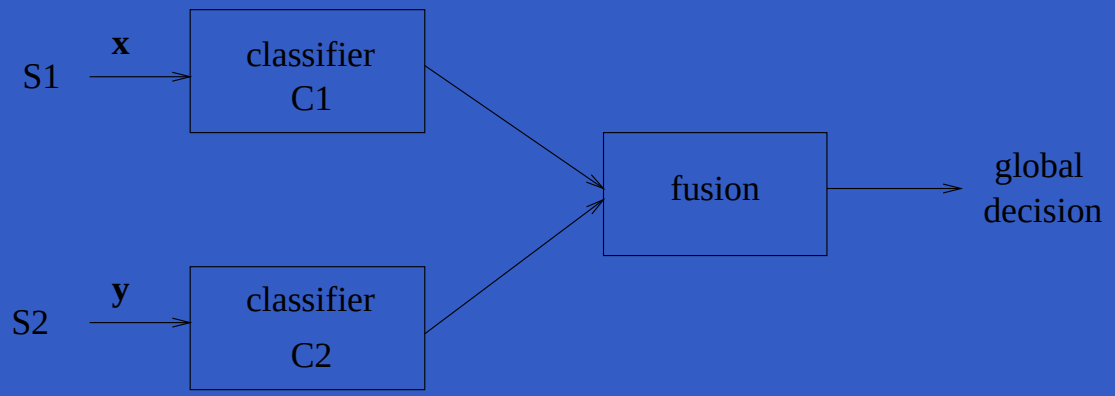
$n = 568$

test : 462 ex.

(different speakers)

Classifier	test error rate
Multi-layer perceptron (88 hidden units)	0.49
Radial Basis Function (528 hidden units)	0.47
Gaussian node network (528 hidden units)	0.45
Nearest neighbor	0.44
Linear Discriminant Analysis	0.56
Quadratic Discriminant Analysis	0.53
CART	0.56
BRUTO	0.44
MARS (degree=2)	0.42
Case-based classifier (33 prototypes)	0.38
Case-based classifier (44 prototypes)	0.37
Case-based classifier (55 prototypes)	0.37

Data fusion example



- $K = 2$ classes
- $\mathbf{x} \in \mathbb{R}^5, \mathbf{y} \in \mathbb{R}^3$, Gaussian distributions, conditionally independent
- Learning set: $n = 60$, cross-validation: $n_{cv} = 100$
- test: 5000 vectors

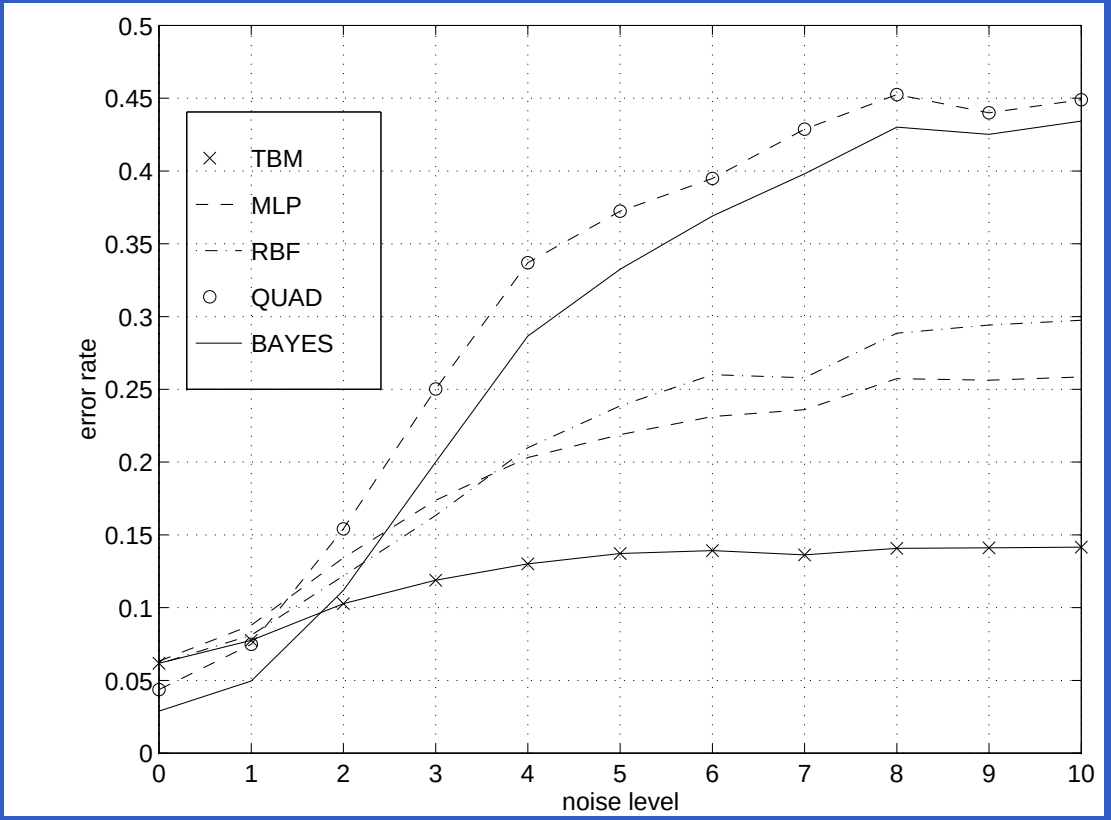
Data fusion: results (1)

Test error rates: uncorrupted data

Method	x alone	y alone	x and y
TBM	0.106	0.148	0.061
MLP	0.113	0.142	0.063
RBF	0.133	0.159	0.083
QUAD	0.101	0.141	0.049
BAYES	0.071	0.121	0.028

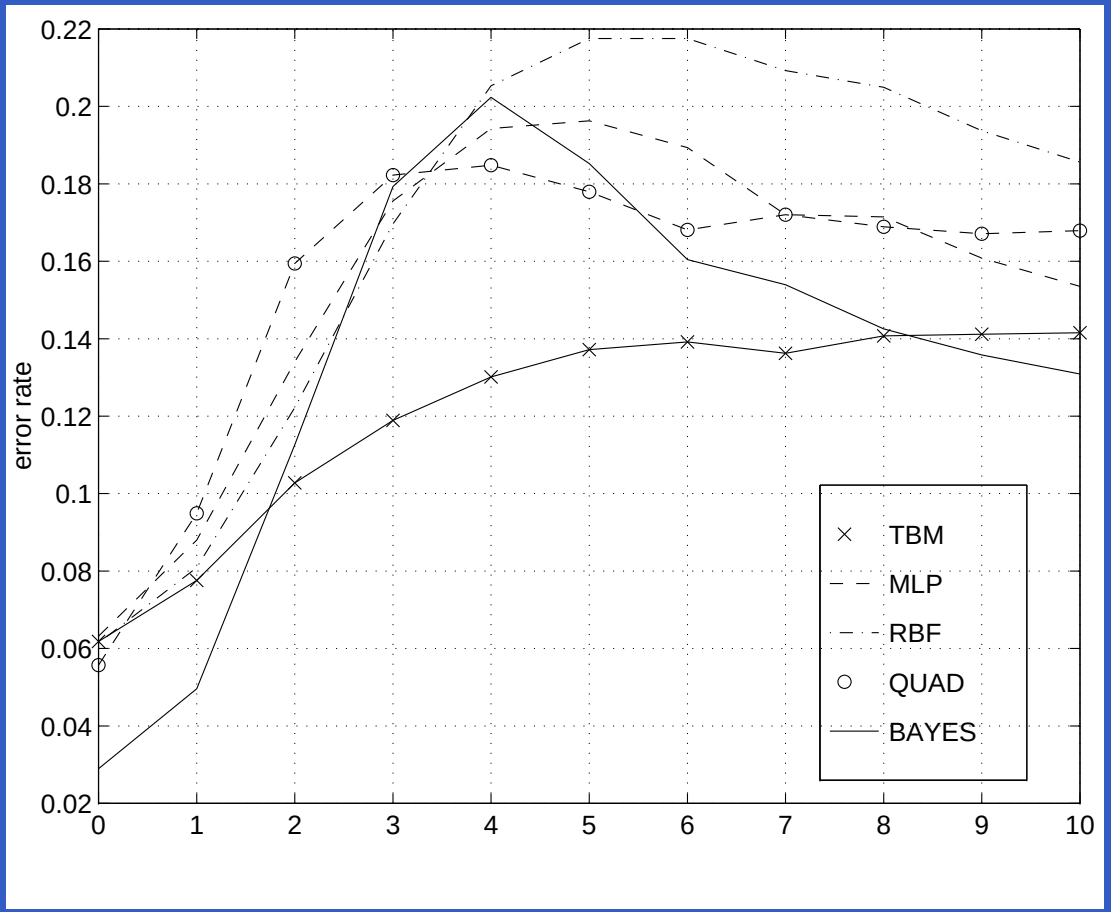
Data fusion: results (2)

Test error rates: $\mathbf{x} + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$



Data fusion: results (3)

Test error rates: $\mathbf{x} + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$, with rejection



Results on EEG data

- $K = 2$ classes, $d = 64$
- data labeled by 5 experts
- $n = 200$ learning patterns, 300 test patterns

k	k-NN	w K-NN	TBM (crisp labels)	TBM (uncert. labels)
9	0.30	0.30	0.31	0.27
11	0.29	0.30	0.29	0.26
13	0.31	0.30	0.31	0.26

Conclusions

- Cased-based classification: a very general paradigm for solving complex pattern recognition problems within the TBM.
- Advantages: possibility to handle imprecise/uncertain training data, robustness and efficiency in classifier fusion problems.
- Simple idea, many possible extensions (adaptive distance, prototypes, etc.).
- The same principle can be applied to regression (prediction of continuous variable), using a fuzzy extension of BF theory.