On belief functions implementations

Arnaud Martin

Arnaud.Martin@univ-rennes1.fr

Univ Rennes, CNRS, IRISA, Lannion, France

Siena, October, 29th 2019

INSTITUT DE RECHERCHE EN INFORMATIQUE ET SYSTEMES ALÉATOIRES

Implementation ways

1. implement the focal elements:
   - in general few focal elements: save space complexity
   - by a list
2. implement the power set:
   - easy to display and understand
   - by a vector

Plan

- Natural order
- Smets codes
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - probabilistic based model
- Classifiers fusion

Plan

- Natural order
- Smets codes
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - probabilistic based model
- Classifiers fusion

Natural order or Binary order

Discernment frame: $\Omega = \{\omega_1, \omega_2, \ldots, \omega_n\}$
Power set: all the disjunctions of Ω:
$2^\Omega = \{\emptyset, \{\omega_1\}, \{\omega_2\}, \{\omega_1 \cup \omega_2\}, \ldots, \Omega\}$

Natural order:

$$
\begin{aligned}
2^\Omega = & \{\emptyset, \{\omega_1\}, \{\omega_2\}, \{\omega_1 \cup \omega_2\}, \\
& \{\omega_3\}, \{\omega_1 \cup \omega_3\}, \{\omega_2 \cup \omega_3\}, \{\omega_1 \cup \omega_2 \cup \omega_3\}, \\
& \{\omega_4\}, \ldots, \Omega\}
\end{aligned}
$$

Natural order or Binary order

Natural order:

$\emptyset$ 0	ω_1 1	ω_2 2	$\omega_1 \cup \omega_2$ $3 = 2^2 - 1$
ω_3 $4 = 2^{3-1}$	$\omega_1 \cup \omega_3$ 5	$\omega_2 \cup \omega_3$ 6	$\omega_1 \cup \omega_2 \cup \omega_3$ $7 = 2^3 - 1$
ω_4 $8 = 2^{4-1}$	$\dots$	$\dots$	$\dots$
ω_i 2^{i-1}	$\dots$	$\dots$	Ω $2^n - 1$

Natural order or Binary order

Bba with R:

Example: $m_1(\omega_1) = 0.5$, $m_1(\omega_3) = 0.4$, $m_1(\omega_1 \cup \omega_2 \cup \omega_3) = 0.1$
$m_2(\omega_3) = 0.4$, $m_2(\omega_1 \cup \omega_3) = 0.6$

```
F1 <- c(1,4,7)
F2 <- c(4,5)
M1 <- c(0.5,0.4,0.1)
M2 <- c(0.4,0.6)
```

Natural order or Binary order

Combination

$$m_{\text{Conj}}(X) = \sum_{Y_1 \cap Y_2 = X} m_1(Y_1) m_2(Y_2) \qquad (1)$$

$\omega_1 \cap (\omega_1 \cup \omega_3)$: $1 \cap 5$
In binary with 3 digits for a frame of 3 elements: 1=100 and 5=101=100 |110
$100 \& 101 = 100$

Natural order or Binary order

With R:

```
sizeDS <- 3
F1 <- c(1,4,7)
F2 <- c(4,5)
M1 <- c(0.5,0.4,0.1)
M2 <- c(0.4,0.6)
FRes=c()
MRes=c()
for (i in 1:length(F1)){
   for (j in 1:length(F2)){
      FRes <- cbind(FRes,bin2dec(dec2bin(F1[i],sizeDS)&dec2bin(F2[j],sizeDS)))
      MRes <- cbind(MRes, M1[i]*M2[j])
   }
}
```

Plan

- Natural order
- Smets codes
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - probabilistic based model
- Classifiers fusion

Plan

- Natural order
- Smets codes
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - probabilistic based model
- Classifiers fusion

Smets codes for DST

Smets gaves the codes of the Mobius transform (see DST_fmt_functions.r) for conversions:

- bba and belief: mtobel, beltom
- bba and plausibility: mtopl, pltom
- bba and communality: mtoq, qtom
- bba and implicability: mtob, btom
- bba to pignistic probability: mtobetp
- etc...

e.g. with R:
source(“./DST_fmt_functions.r”)
m1 <- c(0,0.4,0.1,0.2,0.2,0,0,0.1)
mtobel(m1)
gives: 0 0.4 0.1 0.7 0.2 0.6 0.3 1

For s bbas m_j

Conjunctive combination

$$m_{\mathrm{Conj}}(X) = \sum_{Y_1 \cap \ldots \cap Y_s = X} \prod_{j=1}^{s} m_j(Y_j)$$

The practical way:

$$q(X) = \prod_{j=1}^{s} q_j(X)$$

Disjunctive combination

$$m_{\mathrm{Dis}}(X) = \sum_{Y_1 \cup \ldots \cup Y_s = X} \prod_{j=1}^{s} m_j(Y_j)$$

The practical way:

$$b(X) = \prod_{j=1}^{s} b_j(X)$$

DST code

With R

For the conjunctive rule of combination:

```
m1 <- c(0,0.4,0.1,0.2,0.2,0,0,0.1)
m2 <- c(0,0.2,0.3,0.1,0.1,0,0.2,0.1)
q1 <- mtoq(m1)
q2 <- mtoq(m2)
qConj <- q1*q2
mConj <- qtom(qConj)
mConj
   0.41 0.22 0.2 0.05 0.09 0 0.02 0.01
```

DST code

With R

For the disjunctive rule of combination:

```
m1 <- c(0,0.4,0.1,0.2,0.2,0,0,0.1)
m2 <- c(0,0.2,0.3,0.1,0.1,0,0.2,0.1)
b1 <- mtob(m1)
b2 <- mtob(m2)
bDis <- b1*b2
mDis <- btom(bDis)
mDis
  0 0.08 0.03 0.31 0.02 0.08 0.13 0.35
```

DST code

Once bbas are combined, to decide just use the functions mtobel, mtopl or mtobetp, etc.

With R

```
mtopl(mConj)
  0 0.28 0.28 0.5 0.12 0.39 0.37 0.59
mtobetp(mConj)
  0.4209 0.404 0.1751

mtopl(mDis)
  0 0.82 0.82 0.98 0.58 0.97 0.92 1
mtobetp(mDis)
  0.3917 0.3667 0.2417
```

DST code

DST code for the combination:

- criteria=1 Smets criteria
- criteria=2 Dempster-Shafer criteria (normalized)
- criteria=3 Yager criteria
- criteria=4 disjunctive combination criteria
- criteria=5 Dubois criteria (normalized and disjunctive combination)
- criteria=6 Dubois and Prade criteria (mixt combination)
- criteria=7 Florea criteria
- criteria=8 PCR6
- criteria=9 Cautious Denoeux Min for non-dogmatics functions
- criteria=10 Cautious Denoeux Max for separable functions
- criteria=11 Hard Denoeux for sub-normal functions
- criteria=12 Mean of the bbas
- criteria=13 LNS rule, for separable masses
- criterion=131 LNSa rule, for separable masses (an approximation of LNS)

A. Martin, Conflict management in information fusion with belief functions. E. Bossé; G. Rogova. Information quality in information fusion and decision making, pp.79-97, 2019, Information Fusion and Data Science.

DST code

decisionDST code for the decision:

- criteria=1 maximum of the plausibility
- criteria=2 maximum of the credibility
- criteria=3 maximum of the credibility with rejection
- criteria=4 maximum of the pignistic probability
- criteria=5 Appriou criteria (decision onto 2^{Ω})
- criterion=6 Distance criterion
  A. Essaid, A. Martin, G. Smits, B. Ben Yaghlane, A Distance-Based Decision in the Credal Level, International Conference on Artificial Intelligence and Symbolic Computation (AISC 2014), Dec 2014, Spain.

DST code

```
m1 <- c(0,0.4,0.1,0.2,0.2,0,0,0.1)
m2 <- c(0,0.2,0.3,0.1,0.1,0,0.2,0.1)
m3 <- c(0.1,0.2,0,0.4,0.1,0.1,0,0.1)

m3d <- mydiscounting(m3,0.95)

M_comb_Smets <- DST(cbind(m1,m2,m3d),1)
M_comb_Conj <- DST(cbind(m1,m2,m3d),2)
M_comb_PCR6 <- DST(cbind(m1,m2),8)

class_fusion <- decisionDST(M_comb_Smets,1)
class_fusion <- decisionDST(M_comb_Conj,4)
class_fusion <- decisionDST(M_comb_PCR6,4)
class_fusion <- decisionDST(M_comb_Smets,5,0.5)
```

Plan

- Natural order
- Smets codes
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - Probabilistic based model
- Classifiers fusion

Plan

- Natural order
- Smets codes
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - Probabilistic based model
- Classifiers fusion

Random bbas (see RandomMass.r)

With R:

1. OmegaSize <- 3
2. nbFocalElement <- 4
3. ind <- sample(1:2^OmegaSize)
4. indFocalElement <- ind[1:nbFocalElement]
5. randMass <- diff(c(0, sort(runif(nbFocalElement - 1)), 1))
   We take the difference between 3 ordered random numbers in [0,1]
6. you have to put the mass value on the corresponding focal element in the binary order:
   MassOut <- matrix(0, 2^OmegaSize)
   MassOut[indFocalElement] <- randMass

Random bbas (see RandomMass.r)

- Type=1: focal elements can be evrywhere:
  ind <- Sample(1:2^OmegaSize)
- Type=2: focal elements not on the emptyset:
  ind <- Sample(2:2^OmegaSize)
- Type=3: no dogmatic mass: one focal element is on Omega (ignorance):
  ind <- c(nb, Sample(1:(nb - 1)))
- Type=4: no dogmatic mass: one focal element is on Omega (ignorance) and focal elements are not on the emptyset
  ind <- c(nb, Sample(2:(nb - 1)))
- Type=5: all the focal elements are the singletons:

```
if (nbFocalElement == OmegaSize) {
   ind <- 1 + 2^(1:OmegaSize - 1)
}
```

Distance based model (Denœux 1995)

Only ω_i and Ω are focal elements, $n * m$ sources (experts)

- Prototypes case ($\mathbf{x}_i$ center of ω_i). For the observation x

$$
\begin{aligned}
m_j^i(\omega_i) &= \alpha_{ij} \exp[-\gamma_{ij} d^2(x, \mathbf{x}_i)] \\
m_j^i(\Omega) &= 1 - \alpha_{ij} \exp[-\gamma_{ij} d^2(x, \mathbf{x}_i)]
\end{aligned}
$$

  - $0 \leq \alpha_{ij} \leq 1$: discounting coefficient and $\gamma_{ij} > 0$, are parameters to play on the quantity of ignorance and on the form of the mass functions
  - The distance allows to give a mass to x higer according to the proximity to ω_i
- belief k-nn: we consider the k-nearest neigborhs insteed to $\mathbf{x}_i$
- Then we combine the bbas

With R (from Denœux codes)

See exampleIris.r

```
source("./knn_belief_fusion.r")
iris <- read.table(file="iris.data",sep=",")
train <-
data.matrix(rbind(iris[1:25,1:4],iris[51:75,1:4],iris[101:125,1:4]))
trainLabel <- c(rep(1,25), rep(2,25), rep(3,25))
test <-
data.matrix(rbind(iris[26:50,1:4],iris[76:100,1:4],iris[126:150,1:4]))
trueLabel <- c(rep(1,25), rep(2,25), rep(3,25)) #trainLabel
res <- knn_belief_fusion(train,trainLabel,test,trueLabel,k=3)
```

Probabilistic based model $^{(1/6)}$

- Need to estimate $p(S_j|\omega_i)$
- 2 models proposed by Appriou according to both axioms:
  1. the $n * m$ couples $[M_i^j, \alpha_{ij}]$ are distinct information sources where focal elements are: ω_i, ω_i^c and Ω
  2. If $M_i^j = 0$ and the information is valid $(\alpha_{ij} = 1)$ then it is certain that ω_i is not true.

Model 1: $m_j^i(\omega_i) = M_i^j$
$m_j^i(\omega_i^c) = 1 - M_i^j$

Model 2: $m_j^i(\Omega) = M_i^j$
$m_j^i(\omega_i^c) = 1 - M_i^j$

Adding the reliability α_{ij} with the discounting:
Model 1:

$$\begin{array}{rcl} m_j^i(\omega_i) & = & \alpha_{ij} M_i^j \\ m_j^i(\omega_i^c) & = & \alpha_{ij}(1 - M_i^j) \\ m_j^i(\Omega) & = & 1 - \alpha_{ij} \end{array}$$

Model 2:

$$\begin{array}{rcl} m_j^i(\omega_i) & = & 0 \\ m_j^i(\omega_i^c) & = & \alpha_{ij}(1 - M_i^j) \\ m_j^i(\Omega) & = & 1 - \alpha_{ij}(1 - M_i^j) \end{array}$$

Probabilistic based model $^{(3/6)}$

How to find M_i^j?
3th axiom:

3 Conformity to the Bayesian approach (case where $p(S_j|\omega_j)$ is exactly the reality ($\alpha_{ij} = 1$) for all i, j) and all the *a priori* probabilities $p(\omega_i)$ are known)

Probabilistic based model $^{(4/6)}$

Model 1: $M_i^j = \dfrac{R_j p(S_j|\omega_j)}{1 + R_j p(S_j|\omega_j)}$

$$
\begin{aligned}
m_j^i(\omega_i) &= \frac{\alpha_{ij} R_j p(S_j|\omega_j)}{1 + R_j p(S_j|\omega_j)} \\
m_j^i(\omega_i^c) &= \frac{\alpha_{ij}}{1 + R_j p(S_j|\omega_j)} \\
m_j^i(\Omega) &= 1 - \alpha_{ij}
\end{aligned}
$$

with $R_j \geq 0$ a normalization factor.

Probabilistic based model $^{(5/6)}$

Model 2: $M_i^j = R_j p(S_j|\omega_j)$

$$\begin{aligned} m_j^i(\omega_i) &= 0 \\ m_j^i(\omega_i^c) &= \alpha_{ij}(1 - R_j p(S_j|\omega_j)) \\ m_j^i(\Omega) &= 1 - \alpha_{ij}(1 - R_j p(S_j|\omega_j) \end{aligned}$$

with $R_j \in [0, (\max_{S_j,i}(p(S_j|\omega_j)))^{-1}]$

In practical:

- α_{ij}: discounting coefficient fixed near 1 and $p(S_j|\omega_j)$ can be given by the confusion matrix
- Adapted to the cases where we learn one class against all the others

Probabilistic based model (6/6)

With R: (see exampleIris.r)

- take the previous confusion matrix or mat_conf=[68 12 22 ; 9 42 5 ; 8 2 87]
- mat_mass= bba_type(mat_conf,alpha,model): gives all the possible bbas (*i.e.* number of classes) for the given confusion matrix, alpha (a constant such as 0.95) and the model (1 or 2)
- bbas=buildBbas(res$findLabel,mat_conf,alpha=0.95,model=1): gives the bbas resulting of the founded classes given in findLabel

Probabilistic vs Distance

Difficulties:

- Appriou: learning the probabilities $p(S_j|\omega_j)$
- Denœux: choice of the distance $d(x, \mathbf{x}_i)$

Easiness:

- $p(S_j|\omega_j)$ easier to estimate on decisions with the confusion matrix of the classifiers
- $d(x, \mathbf{x}_i)$ easier to choose on the numeric outputs of classifiers (ex.: Euclidean distance)

Plan

- Natural order
- Smets codes
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - Probabilistic based model
- Classifiers fusion

Plan

- Natural order
- Smets codes
- How to obtain bbas?
  - Random bbas
  - Distance based model
  - Probabilistic based model
- Classifiers fusion

Fusion of classifiers

We consider four classifiers

Classical knn - inverse distance (betweeen 0 and 1) output with each class
res1 <- knn_fusion(train,trainLabel,test,trueLabel,k=3)

Fuzzy knn - fuzzy degree (betweeen 0 and 1) output on each class
res2 <- knn_fuzzy_fusion(train,trainLabel,test,trueLabel,k=3,kf=5)

Bayes classifier - probabilities output on each class
res3 <- bayes_classifier(train,trainLabel,test,trueLabel)

Evidential knn - bbas output with only sigleton and ignorance
res4 <- knn_belief_fusion(train,trainLabel,test,trueLabel,k=3)

Fusion of classifiers: step discounting

We have to discount outputs on singleton (probabilities, distance and fuzzy) in order to add ignorange

```
source(“./mydiscounting.r”)
alpha <- 0.5
mass_knn_s <- mydiscounting(res1$matr_sort_classif,alpha)
mass_knnf_s <- mydiscounting(res2$matr_sort_classif,alpha)
mass_BC_s <- mydiscounting(res3$matr_sort_classif,alpha)
```

Exercice: code mydiscounting.r

Solution: code mydiscounting.r

```
mydiscounting <- function (m,alpha){
  masse <- m*alpha
  masse <- rbind(masse,1-alpha)
  return(masse) }
```

Fusion of classifiers: step converion in binary order

In order to use the library ibelief we have to convert the vector in binary order:

```
source("./conversionBinaryOrder.r")
mass_knn <- conversionBinaryOrder(mass_knn_s)
mass_knnf <- conversionBinaryOrder(mass_knnf_s)
mass_BC <- conversionBinaryOrder(mass_BC_s)
mass_eknn <- conversionBinaryOrder(res4$matr_sort_classif)
```

Exercice: code conversionBinaryOrder.r

Solution: code conversionBinaryOrder.r

```
conversionBinaryOrder <- function (massIn){
  nbClass<-dim(massIn)[1]-1    last<- 2^nbClass
  massOut <- matrix(0,nrow=last,ncol=dim(massIn)[2])
  for (i in 1:nbClass)
    massOut[2^(i-1)+1,] ¡- massIn[i,]
  massOut[last,] <- massIn[nbClass+1,]
  return(masseOut)
}
```

Exercices

- Compare the results before and after fusion
- Find the best combination rule
- Find the best decision rule
- Change the dataset: try yours!

References

- Toolboxes: on http://www.bfasociety.org
  iBelief: R package:
  https://cran.rstudio.com/web/packages/ibelief/index.html
- a lot of papers on: http://www.bfasociety.org

References

- On the presented codes:

  - Kennes R. and Smets Ph. (1991) Computational Aspects of the Möbius Transformation. Uncertainty in Artificial Intelligence 6, P.P. Bonissone, M. Henrion, L.N. Kanal, J.F. Lemmer (Editors), Elsevier Science Publishers (1991) 401-416.
  - A. Martin, Implementing general belief function framework with a practical codification for low complexity, in Advances and Applications of DSmT for Information Fusion, American Research Press Rehoboth, pp. 217-273, 2009.
  - T. Denœux. A k-nearest neighbor classification rule based on Dempster-Shafer theory. IEEE Transactions on Systems, Man and Cybernetics, 25(05):804-813, 1995.
  - L. M. Zouhal and T. Denœux. An evidence-theoretic k-NN rule with parameter optimization. IEEE Transactions on Systems, Man and Cybernetics - Part C, 28(2):263-271,1998.
  - Appriou, Discrimination multisignal par la théorie de l'évidence, chap 7, Décision et Reconnaissance des formes en signal, Hermes Science Publication, 2002, 219-258

References

- Other way to code belief functions:

  - P.P. Shenoy and G. Shafer. Propagating belief functions with local computations. IEEE Expert, 1(3):43-51, 1986.
  - R. Haenni and N. Lehmann. Implementing belief function computations. International Journal of Intelligent Systems, Special issue on the Dempster-Shafer theory of evidence, 18(1):31-49, 2003.
  - C. Liu, D. Grenier, A.-L. Jousselme, É. Bossé, Reducing algorithm complexity for computing an aggregate uncertainty measure, IEEE Transactions on Systems, Man and Cybernetics-Part A: Systems and Humans 37: 669-679, 2007.
  - V.-N. Huynh, Y. Nakamori, Notes on "Reducing Algorithm Complexity for Computing an Aggregate Uncertainty Measure", IEEE Transactions on Cybernetics-Part A: Systems and Humans 40: 205-209, 2010.
  - M. Grabisch. Belief functions on lattices. International Journal of Intelligent Systems, 24:76-95, 2009.