Dempster-Shafer Belief Functions

Combining Belief Functions

Types of Belief Functions

Belief Functions in Expert Systems
Belief Functions

The standard text for definitions, etc. is, of course:

A belief function on a frame $\Theta$ is a function $\text{Bel} : 2^\Theta \rightarrow [0, 1]$ such that:

1. $\text{Bel}(\emptyset) = 0$
2. $\text{Bel}(\Theta) = 1$
3. $\text{Bel}(A_1 \cup \ldots \cup A_n) \geq \sum_i \text{Bel}(A_i) - \sum_{i<j} \text{Bel}(A_i \cap A_j) + \ldots + (-1)^{n+1} \text{Bel}(A_1 \cap \ldots \cap A_n)$

Plausibility is defined by $\text{Pl}(A) = 1 - \text{Bel}(\neg A)$
Belief Functions

*Basic probability assignments are functions* $m : 2^\Theta \to [0, 1]$ *such that:*

1. $m(\emptyset) = 0$
2. $\sum_{A \subseteq \Theta} m(A) = 1$

*Then we may define* $\text{Bel}(A) = \sum_{B \subseteq A} m(B)$
Example:

* Consider a frame with three possible outcomes \( \{a, b, c\} \)

* Suppose we are given the following basic probability assignment:

\[
\begin{align*}
m(a) &= .1; m(b) = .1; m(c) = .1; \\
m(a, b) &= .1; m(a, c) = .2; m(b, c) = .3; \\
m(a, b, c) &= .1
\end{align*}
\]
### Belief Functions

<table>
<thead>
<tr>
<th></th>
<th>Bpa</th>
<th>Bel</th>
</tr>
</thead>
<tbody>
<tr>
<td>∅</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>{a}</td>
<td>.1</td>
<td>.1</td>
</tr>
<tr>
<td>{b}</td>
<td>.1</td>
<td>.1</td>
</tr>
<tr>
<td>{c}</td>
<td>.1</td>
<td>.1</td>
</tr>
<tr>
<td>{a,b}</td>
<td>.1</td>
<td>.3</td>
</tr>
<tr>
<td>{a,c}</td>
<td>.2</td>
<td>.4</td>
</tr>
<tr>
<td>{b,c}</td>
<td>.3</td>
<td>.5</td>
</tr>
<tr>
<td>{a,b,c}</td>
<td>.1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Bpa</td>
<td>Bel</td>
</tr>
<tr>
<td>--------</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>$\emptyset$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>{a}</td>
<td>.1</td>
<td>.1</td>
</tr>
<tr>
<td>{b}</td>
<td>.1</td>
<td>.1</td>
</tr>
<tr>
<td>{c}</td>
<td>.1</td>
<td>.1</td>
</tr>
<tr>
<td>{a,b}</td>
<td>.1</td>
<td>.3</td>
</tr>
<tr>
<td>{a,c}</td>
<td>.2</td>
<td>.4</td>
</tr>
<tr>
<td>{b,c}</td>
<td>.3</td>
<td>.5</td>
</tr>
<tr>
<td>{a,b,c}</td>
<td>.1</td>
<td>1</td>
</tr>
</tbody>
</table>
Belief Functions

Bpas may be recovered from Bel functions using

\[ m(A) = \sum_{B \subseteq A} (-1)^{|A - B|} \text{Bel}(B) \]
Belief Functions

The *commonality* function is a function

\[ Q : 2^\Theta \rightarrow [0, 1] \]

defined by

\[ Q(A) = \sum_{A \subseteq B} m(B) \]

Bpas may be recovered from commonality functions using

\[ m(A) = \sum_{A \subseteq B} (-1)^{|B - A|} Q(B) \]
## Belief Functions

<table>
<thead>
<tr>
<th></th>
<th>Bpa</th>
<th>Bel</th>
<th>Pl</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>∅</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>{a}</td>
<td>.1</td>
<td>.1</td>
<td>.5</td>
<td>.5</td>
</tr>
<tr>
<td>{b}</td>
<td>.1</td>
<td>.1</td>
<td>.6</td>
<td>.6</td>
</tr>
<tr>
<td>{c}</td>
<td>.1</td>
<td>.1</td>
<td>.7</td>
<td>.7</td>
</tr>
<tr>
<td>{a,b}</td>
<td>.1</td>
<td>.3</td>
<td>.9</td>
<td>.2</td>
</tr>
<tr>
<td>{a,c}</td>
<td>.2</td>
<td>.4</td>
<td>.9</td>
<td>.3</td>
</tr>
<tr>
<td>{b,c}</td>
<td>.3</td>
<td>.5</td>
<td>.9</td>
<td>.4</td>
</tr>
<tr>
<td>{a,b,c}</td>
<td>.1</td>
<td>1</td>
<td>1</td>
<td>.1</td>
</tr>
</tbody>
</table>
Belief Functions

- Recall that the bpa function can be uniquely recovered from Pl, Bel or Q
- In fact, we can convert any one of the four representations uniquely into any of the others
- These conversions are examples of Möbius transforms
- There are Fast Möbius Transforms to do this efficiently (see Kennes paper)
Belief Functions
In expert systems based on belief functions:

- user inputs are often in the form of bpas
- propagation is most efficient implemented via commonalities
- marginalization is most efficient implemented via Bel functions
- output is often desired as Bel or Pl functions
Combining Belief Functions

Dempster’s Rule

Consider two belief functions given by their bpas as follows:

\[
\begin{align*}
\text{m}_1 (\{a\}) &= 0.5; \\
\text{m}_1 (\{\sim a\}) &= 0.3; \\
\text{m}_1 (\{a, \sim a\}) &= 0.2; \\
\text{m}_2 (\{a\}) &= 0.7; \\
\text{m}_1 (\{\sim a\}) &= 0.2; \\
\text{m}_1 (\{a, \sim a\}) &= 0.1
\end{align*}
\]
## Combining Belief Functions

<table>
<thead>
<tr>
<th></th>
<th>{a}</th>
<th>{\sim a}</th>
<th>{a,\sim a}</th>
</tr>
</thead>
<tbody>
<tr>
<td>(m_1)</td>
<td>0.5</td>
<td>0.3</td>
<td>0.2</td>
</tr>
</tbody>
</table>
| \(m_2\) | \begin{align*} & \{a\} \\
& 0.7 \quad 0.7 \times 0.5 = 0.35 \\
& \{\sim a\} - \\
& \{a,\sim a\} \end{align*} | \begin{align*} & \{a\} \\
& 0.2 \quad 0.2 \times 0.5 = 0.10 \\
& \{\sim a\} - \\
& \{a,\sim a\} \end{align*} | \begin{align*} & \{a\} \\
& 0.1 \quad 0.1 \times 0.5 = 0.05 \\
& \{\sim a\} - \\
& \{a,\sim a\} \end{align*} |

\[
m_1 \otimes m_2 (\{a\}) = \frac{0.35 + 0.14 + 0.05}{1 - (0.21 + 0.10)} = \frac{0.54}{0.69} = 0.783
\]

\[
m_1 \otimes m_2 (\{a,\sim a\}) = \frac{0.02}{1 - (0.21 + 0.10)} = 0.029
\]

\[
m_1 \otimes m_2 (\{\sim a\}) = \frac{0.06 + 0.04 + 0.03}{1 - (0.21 + 0.10)} = \frac{0.13}{0.69} = 0.188
\]
Note, however, the following:

<table>
<thead>
<tr>
<th></th>
<th>$m_1$</th>
<th>$Q_1$</th>
<th>$m_2$</th>
<th>$Q_2$</th>
<th>$Q_1 \times Q_2$</th>
<th>$m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>{a}</td>
<td>.5</td>
<td>.7</td>
<td>.7</td>
<td>.8</td>
<td>.56</td>
<td>.54</td>
</tr>
<tr>
<td>{~a}</td>
<td>.3</td>
<td>.5</td>
<td>.2</td>
<td>.3</td>
<td>.15</td>
<td>.13</td>
</tr>
<tr>
<td>{a,~a}</td>
<td>.2</td>
<td>.2</td>
<td>.1</td>
<td>.1</td>
<td>.02</td>
<td>.02</td>
</tr>
</tbody>
</table>

After normalization, these are the same values as derived from Dempster’s Rule.
Combining Belief Functions

In expert system applications, therefore, it is efficient to:

* use Fast Möbius Transforms to convert bpas to commonalities
* combine the commonalities by pointwise multiplication
* (eventually) use Fast Möbius Transforms to convert the results back to bpas or other desired outputs
Types of Belief Functions

- If $A$ is a subset of the frame $\Theta$ of a belief function, then $A$ is a *focal element* if $m(A) > 0$.
- The *core* of a belief function is the union of all its focal elements.
- If, for some subset $A$, $m(A) = s$ and $m(\Theta) = 1 - s$ then $m$ is a *simple support function*.
- Thus a *simple support function* has only one focal element other than the frame itself.
A belief function that is the combination of one or more simple support functions is called a *separable support function*.

A belief function that results from marginalizing a separable support function may not itself be separable; it is called a *support function*; Shafer suggests these are fundamental for the representation of evidence.
Types of Belief Functions

- Simple support functions
  - Separable support functions
  - Support functions
  - Belief functions

- A belief function whose focal elements are nested is called a consonant belief function
A belief function that is not a support function is called a quasi support function.

Quasi support functions arise as the limits of sequences of support functions.

A belief function for which $\text{Bel}(A \cup B) = \text{Bel}(A) + \text{Bel}(B)$ whenever $A \cap B = \emptyset$ is called a Bayesian belief function.

Equivalently, a Bayesian belief function is a belief function all of whose focal elements are singletons.

Bayesian belief functions are quasi support functions (except when $\text{Bel} \left( \{\theta\} \right) = 1$ for some $\theta \in \Theta$).
Belief Functions in Expert Systems

- Belief functions can be propagated locally in Join Trees (Markov Trees) using the Shenoy-Shafer algorithm.
- Belief functions can also be propagated locally in Junction Trees using the Aalborg architecture; this requires division (of commonalities) and intermediate results may not be interpretable.
- In practice, it is most efficient to perform combination using commonalities and marginalization using Bels.
Belief Functions in Expert Systems

- Xu and Kennes give efficient algorithms for carrying out belief function combination, for bit-array representations of subsets, and for Fast Möbius Transforms.
- The bit-array representation includes algorithms for testing subsets, forming intersections, unions, etc directly with the bit-arrays.
- Full details of the Fast Möbius Transform algorithms are given in Kennes.
Belief Functions in Expert Systems

- Efficient implementations are especially important for belief functions
  - $n$ binary variables generate a joint space with $2^n$ configurations in probability systems
  - $n$ binary variables generate a joint space with $2^{2^n}$ potential focal elements in belief function systems
Belief Functions in Expert Systems

“AND” nodes can be defined in belief function terms

* Suppose we wanted to create a relationship showing that a variable A is true iff variables B and C are both true
* In a Bayesian network, we could use:

\[
\begin{array}{c|c}
| a, b, c | \quad 1 \\
| a, b, \sim c | \quad 0 \\
| a, \sim b, c | \quad 0 \\
| a, \sim b, \sim c | \quad 0 \\
| \sim a, b, c | \quad 0 \\
| \sim a, \sim b, c | \quad 1 \\
| \sim a, \sim b, \sim c | \quad 1 \\
\end{array}
\]
Belief Functions in Expert Systems

- “AND” nodes can be defined in belief function terms
  - Suppose we wanted to create a relationship showing that a variable A is true iff variables B and C are both true
  - What would we use for belief functions?
Belief Functions in Expert Systems

“AND” nodes can be defined in belief function terms

* Suppose we wanted to create a relationship showing that a variable A is true iff variables B and C are both true
* What would we use for belief functions?

\[
\left\{ (a, b, c), (~ a, b, ~ c), (~ a, ~ b, c), (~ a, ~ b, ~ c) \right\} | 1
\]
Discounted “AND” nodes can also be defined

* Suppose we want A to be certain if B and C are both certain, but B and C both to be true with probability 0.95 when A is certain

\[
\begin{pmatrix}
1 & 0 & 0.0526 \\
0 & 0 & 0 \\
0 & 0 & 1 \\
0 & 1 & 1 \\
0 & 1 & 0.9474
\end{pmatrix}
\]
Discounted “AND” nodes can also be defined

* Suppose we want A to be certain if B and C are both certain, but B and C both to be true with bpa 0.95 when A is certain

\[
\begin{bmatrix}
\{(a,b,c), (~ a,b,~ c), (~ a,~ b,c), (~ a,~ b,~ c)\} & 0.95 \\
\{(a,b,c), (a,~ b,~ c), (~ a,b,~ c), (~ a,~ b,c), (~ a,~ b,~ c)\} & 0.05
\end{bmatrix}
\]
Shafer & Srivastava show how to apply mean-per-unit sampling using belief functions

Gillett & Srivastava show how to perform attribute sampling using belief functions

Gillett shows how to apply monetary unit sampling using belief functions
Belief Functions in Expert Systems

- Elicitation of bpas from domain experts is potentially more difficult even than for probabilities, partly because of unfamiliarity, but more importantly because far more parameters need to be obtained.
- Eliciting expert beliefs in a sufficiently general way that they can be interpreted as either probabilities or bpas for comparative studies is even trickier!
Belief Functions in Expert Systems

One possibility

- Elicit two parameters
  - The ratio $f$ estimating how much more support the evidence provides for the objective than against it
  - The degree of indeterminacy $i$ estimating the extent to which the evidence fails to provide persuasive evidence for or against the objective
Belief Functions in Expert Systems

One possibility

* For probabilities

\[
\begin{bmatrix}
o \\
\sim o \\
o \\
\end{bmatrix}
= \begin{bmatrix}
\frac{f}{1+f} \\
1 \\
\frac{1}{1+f} \\
\end{bmatrix}
\]
Belief Functions in Expert Systems

- One possibility

* For belief functions

\[
\begin{bmatrix}
o & \frac{f-i}{1+f} \\
\sim o & \frac{1-f \times i}{1+f} \\
o, \sim o & i
\end{bmatrix}
\]
Belief Functions in Expert Systems

- As in the case of probabilities, joint valuations cannot be uniquely determined from marginals (which is often all domain experts provide).
- Depending on the application, however, “best” or “worst” cases can sometimes be identified.
Belief Functions in Expert Systems

- The Shafer & Srivastava paper we read for today sets out extensive arguments why belief functions might be considered superior to probabilities for certain applications, such as auditing.

- Among these reasons, the one that first attracted me to study belief functions when I was building an Expert System is the argument that they better represent ignorance.

- In auditing, for example, accounts receivable, insufficient replies from customers might lead us to assess a probability of, say, only 70% that accounts receivable exist.

- Probability theory then forces us to assess a 30% probability that they do not exist, despite the fact that there is no evidence they do not - merely insufficient evidence that they do.
Belief Functions in Expert Systems

- Belief functions allow us to assign a 70% bpa to existence, and the balance to the whole frame, representing ignorance.
- In probability theory there would be no difference if some of the missing customers in fact wrote to deny the existence of the balance.
- Using belief functions, however, we could assign some part of the bpa to represent contrary evidence, and the remainder to ignorance - perhaps \( m(\text{exist}) = 0.7; m(\neg \text{exist}) = 0.2; m(\text{exist, } \neg \text{exist}) = 0.1 \).
- Of course, in belief function terms, complete ignorance is represented by \( m(\text{exist}, \neg \text{exist}) = 1 \): it must be one of the outcomes, we don’t know which, or which is more likely.
- Probabilistically, ignorance is represented as \( P(\text{exist}) = P(\neg \text{exist}) = 0.5 \) and we have to assume the outcomes equally likely.