FLEXIBLE DATA FUSION (& FISSION)

Alexander Yeh

MIT Laboratory for Computer Science, 545 Technology Sq., Cambridge, MA 02139

ABSTRACT*

An approach is described for developing methods for "data fusion": given how events A & B occurring by themselves influence some measure, estimate the influence (on that measure) of A and B occurring together. An example is "combine the effects of evidence on the belief (likelihood) of some hypothesis." This approach also deals with the opposite problem of estimating the effects on a measure of A and B by themselves when only their combined effects are known: *data fission*. The methods developed will both 1) try to make intuitive estimations at information not given, and 2) not conflict with any information given (unless it is inconsistent).

1. INTRODUCTION

An approach is described for developing methods for "data fusion": given how events A & B occurring by themselves influence some measure, estimate the influence (on that measure) of A and B occurring together. Examples include: 1) combine the effects of evidence on the belief (likelihood) of some hypothesis, and 2) combine the effects of pollutants on public health measures. The opposite problem of estimating the effects on a measure of A and B by themselves when only their combined effects are known, *data fission*, is also discussed.

Data fusion problems can be characterized by two properties:

1. In combining events' influences, a whole range of answers consistent with the given information exists.

2. Often, one or a few intuitive ways exist to combine the influences which tends to "work."

For example: one wants P[A&B] (the probability of A and B occurring), but only has P[A] and P[B]. Then P[A&B] can range from 0 (the two are mutually exclusive) to min[P[A],P[B]] (A implies B or vice-versa). Furthermore, given that A & B are "typical" events, one might guess that P[A&B] = P[A]P[B] (A and B are independent, a case between the two extremes).

Past efforts in this area include fuzzy sets and logic [5], the Dempster-Shafer rule of combination (orthogonal sum) [7], simplified rules of probability in PROSPECTOR [3], and the certainty factor idea in MYCIN [8]. They have concentrated on intuitive methods of combination, figuring rightly that asking people giving knowledge (to the system) to give the effects of every combination of events would be impossible. However, with these efforts, when the "intuitive" method fails on some case, and the correct answer is known, there is no clean way to tell the system. Often, one misrepresents some value(s) in the system so that when they combine, they more or less give the correct answer. For example, let the MYCIN interpreter have the two rules

1. Event A is evidence for event C: CF[C,A] = 0.3 (a positive value indicates evidence in favor, a negative value indicates evidence against).

2. But, both events A and B occurring is evidence for C not occurring: CF[C,A&B] = –0.3.

Now when both A and B occur, both rules will be active, and the net evidence for C will be the two figures added together (0.3–0.3 = 0) and not –0.3, the intended figure. To fix this, one might make CF[C,A&B] = –0.6. For a further discussion of these systems, see [9].

To avoid this problem, methods for data fusion (and fission) should supply both the intuitive answers for combinations lacking more specific data, and the appropriate answers for known combinations.

2. GENERAL APPROACH

The following is a general approach for developing a data fusion/fission system:

1. Determine the form(s) of information you and others want to give the system (it should be easy to provide), and the form(s) the system should be able to output.

2. Determine the realm of all legal possibilities for combinations of given and requested information. That is, determine when a group of given statements (information) should be considered consistent.

3. Determine a criterion (rule of thumb) to make estimations when information given to the system is not specific (discussed below).

4. Find a specific method (implementation) that:

   a. treats the given information as constraints on the possible results, and reports inconsistencies.

   b. uses the rule of thumb to find the estimate when

*This report describes work done at the MIT Laboratory for Computer Science. It was supported in part by the National Institute of Health, under contract # NIH-1-R01-HL33041-01.

the given information is not specific enough to "force" a result. Limit the result to one that is consistent with the supplied information.

c. does not require too much storage. A simple way make all system estimates overridable is to always store an overwritable estimate for each possible combination of events to be considered. However, often an exponential number of such combinations exist.

A method meeting these criteria 1) provides estimates where precise information is lacking, and 2) permits the use of precise information when it is available.

These steps may have to be iterated a few times because 1) the information and/or criterion in the first three steps turn out to be not quite what you want, or 2) a specific implementation eludes you in step 4.

Some heuristics for finding a "rule of thumb" in step 3 are the following:

1. It will 'typically' give the correct answer or something close. If you find such a rule of thumb, this heuristic takes the highest priority.

2. When given a range of choices, it will tend towards the middle ground (as opposed to the extremes). This way it will not be too far wrong.

3. It is simple conceptually (maybe easier to debug and anticipate effects).

An often applicable rule of thumb for fusion is *non-interaction* between entities. Examples include assumptions of independence in probability (and similarly, maximizing entropy, see [9]), and linearity (the total is just the sum of the parts) in many fields. Justifications for using this include:

1. users (people) tend to notice and mention strong interactions (such as *A causes B*, or *seeing C indicates not D*, etc.), so what tends not to be explicitly given are the non-interactions,

2. it is in the middle between positive and negative interaction,

3. it is more conceptually "simple" than interaction.

An often applicable rule of thumb for fission is to divide evenly, and as if things did not interact. Justifications for the former are

1. users tend to notice and mention inequities, so what is left tends to be equitably distributed,

2. it is in between favoring any one thing.

Justifications for non-interaction are the same as the ones in fusion.

3. EXAMPLE: LIKELIHOOD INFERENCE

The following is an example of developing a system with the above approach. The system infers likelihoods of events. Information is given on how likely the events are when certain evidence occurs, and on what evidence exists. The example is for the most part from work done by Konolige for PROSPECTOR [4] and extended by Yeh [9], and the details may be found there.

Step 1: People will give the system information of the form *[some term 1]* ??? *[some term 2]*, where

- ??? is $=, <, >, \leq$, or $\geq$, and
- *[some term α]* (α = 1 or 2) is an *a priori* or conditional probability, a MYCIN certainty factor, a correlation, or a numeric constant.

Examples include P[A or B] $\geq$ 0.5, *probability of A when B occurs* $\equiv$ P[A|B] < P[A], *the certainty factor of A given B* $\equiv$ CF[A,B] = 0.3 and correlation(A,B) = correlation(B,C), where A, B, and C are events.

CF[A,B] measures how much knowing B occurs tells you how likely A occurs, and ranges from 1 (B→A) to −1 (B→~A). CF[A,B] = 0 means that knowing B tells you nothing about A. CF[A,B] can be translated into probabilistic terms (see [9]). For example, CF[A,B] = 0.3 = (P[A|B]−P[A])/(1−P[A]).

Correlation(A,B) measures how much knowing A or B occurs tells you how likely the other will occur, and also ranges from 1 (B↔A) to −1 (B→~A and vice-versa). A value of 0 means that A and B are independent. Correlation(A,B) is defined in terms of probabilities as

$$(P[A,B]-P[A]P[B])/(P[A](1-P[A])P[B](1-P[B]))^{0.5}.$$

The system will output information of the same form as the input.

Step 2: All of the input and output statements can be put in terms of probabilities. So, there should be at least one common probability distribution in which all of the statements hold. If not, the statements will describe a situation which is impossible given the laws of probability.

Step 3: The rule of thumb used is *maximize the independence between events*. The reasons for using these are the same as the ones mentioned in the last section.

Step 4: The method the system will use is to find the probability distribution which minimizes the sum of squares $\Sigma_i p_i^2$ (it is like maximizing entropy, and will range from 2^{-n} to 1.0), where n is the number of events, and p_i is the probability of the ith combination of events occurring and not occurring out of the 2^n possible combinations of the distribution (example, if the events are A and B, p_i can be one of ~A&~B, ~A&B, A&~B, or A&B). Justifications are in [9].

As described, the problem can viewed as "pick the values (for the probability distribution) that minimizes some function, while satisfying certain constraints (are consistent with the input information and indeed form a probability distribution). If none can satisfy the constraints, report that." In the operations

research field, this is called a (non-linear) programming problem, and many numeric computer programs have been developed to solve them (example: the Simplex algorithm). The type of algorithm to be used for this method is one used for generalized geometric programming (GGP) [1, 2]. The specific algorithm used is discussed in [6].

Some warnings about these algorithms: From a given starting point, they may not converge (even when a solution exists), or converge to a point other than the true global minimum. To some extent, using many random starting points will ameliorate this. For GGP algorithms, combination probabilities have to be strictly > 0. In [6], an error exists: $\Sigma\mu(g'(t)-1)$ in the Lagrangian function should be $-\Sigma\mu(g'(t)-1)$. Also, [6]'s algorithm uses exponentiation to find the approximate solution. To help prevent over/underflows, force that exponentiation to return numbers within "reasonable" bounds (examples: all variables are $\geq 10^{-10}$, all combination probabilities are ≤ 1.0).

After the distribution has been found by this method (which can involve a fair amount of computation), the system can use the distribution to fairly quickly answer various questions that may be ask of it (give output statements of the form mentioned in step 1).

An example of setting up a problem for a programming method: The events are A, B, & C, and the input statement is CF[C,A] = 0.3. First, translate it into a constraint on the probabilities of the 2^3 combinations of the events:

$$(P[C,A]/P[A] - P[C])/(1-P[C]) = 0.3,$$

where P[C,A] = P[A,~B,C] + P[A,B,C], and similarly for P[A] and P[C]. Now, use the method in [6] to find the values of the combination probabilities of the distribution which minimize

$$P[\sim A,\sim B,\sim C]^2 + P[\sim A,\sim B,C]^2 + P[\sim A,B,\sim C]^2 + P[\sim A,B,C]^2 + P[A,\sim B,\sim C]^2 + P[A,\sim B,C]^2 + P[A,B,\sim C]^2 + P[A,B,C]^2$$

subject to the constraint above, plus the constraints that any distribution for A, B, & C should obey (> 0 is used instead of ≥ 0 because of GGP):

$$P[\sim A,\sim B,\sim C],\ P[\sim A,\sim B,C],\ P[\sim A,B,\sim C],\ \ldots,\ P[A,B,C] > 0$$

$$P[\sim A,\sim B,\sim C] + P[\sim A,\sim B,C] + P[\sim A,B,\sim C] + \ldots + P[A,B,C] = 1.$$

The system returns the distribution's combination proabilities (rounded to 4 digits, order as in the sum of squares function above):

0.1553 0.1166 0.1553 0.1166
0.0729 0.1553 0.0729 0.1553

From this distribution, one finds that P[A,B] = P[A]×P[B], P[B,C] = P[B]×P[C], and (P[C|A,B]-P[C])/(1-P[C]) = CF[C,A&B] = 0.3 = CF[C,A]. Here, no relations were given between B and either A or C, so the system assumes that B has no effect on (is independent of) either A or C. If later, one learns that CF[C,A&B] = -0.3 = (P[C|A,B]-P[C])/P[C] (positive and negative CF's have different forms), one can add this constraint, and the system will return the distribution (order as before):

0.1589 0.1380 0.1589 0.1380
0.0114 0.2068 0.1148 0.0732

which satisfies both CF constraints.

Since n events leads to a 2^n element distribution, trying to find the one joint distribution underlying all events is a problem. To lower the storage requirements, [9] describes ways to organize the events to take advantage of forms of independence between them. These forms let one find probabilities involving two or more sets of events jointly from probabilities involving those sets separately. A very simple example: if the events A,B are independent from the events D,E, then P[*a,b,c,d*] = P[*a,b*]P[*d,e*], where *a* is A or ~A, *b* is B or ~B, etc. This lets the 4 event distribution be found from two 2 event distributions.

4. SUMMARY

An approach for developing flexible data fusion and fission methods is presented. Such methods will both 1) try to make intuitive estimations at information not given, and 2) not conflict with any information given (unless it is inconsistent). Past work has concentrated on the intuitive estimation for "fusion" aspect, and more or less ignored the nonconflict aspect. An example of using this approach to develop a method for likelihood inference is given. The method has yet to be fully implemented, so doing this, as well as trying the approach on new examples remain as possible work for the future.

REFERENCES

[1] edited by M. Avriel. *Advances in Geometric Programming*. Plenum Press, New York, 1980.

[2] Beightler, C. and Phillips, D. *Applied Geometric Programming*. John Wiley and Sons, Inc., 1976.

[3] Duda, R., Hart, P., and Nilsson, N. "Subjective Bayesian methods for rule-based inference systems." In *AFIPS Conference Proceedings, Vol. 45*, pages 1075-1082. National Computer Conference, 1976.

[4] Duda, R., Hart, P., Konolige, K., Reboh, R. *A Computer Based-Consultant for Mineral Exploration*. Final Report, SRI Project 6415, SRI International, Menlo Park, CA, 1979.

[5] Gaines, B. "Foundations of Fuzzy Reasoning." *International Journal of Man-Machine Studies*. 8(6):623-668, Nov., 1976.

[6] Rijckaert, M. & Martens, X. "A Condensation Method for Generalized Geometric Programming." *Mathematical Programming* 11:89-93, 1976.

[7] Shafer, G. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, 1976.

[8] Shortliffe, E., and Buchanan, B. "A Model of inexact reasoning in medicine." *Mathematical Biosciences* 23:351-379, 1975.

[9] Yeh, A. "PLY: A System of Plausibility Inference with a Probabilistic Basis." Master's thesis, MIT, Cambridge, MA, June, 1983.