Scoring Hypotheses from Threat Detection Technologies: Analogies to Machine Learning Evaluation

Robert C. Schrag[*] and Masami Takikawa[†]

[*]Global InfoTek, Inc., 1920 Association Dr, Suite 200, Reston, VA 20191 USA, rschrag@globalinfotek.com
[†]Cleverset, Inc., 673 NW Jackson Ave, Corvallis, OR 97330 USA, takikawa@cleverset.com
The authors were employed by Information and Transport, Inc. (IET) when this work was performed.

Abstract

We have developed efficient methods to score structured hypotheses from technologies that fuse evidence from massive data streams to detect threat phenomena. We have generalized metrics (precision, recall, F-value, and area under the precision-recall curve) traditionally used in the information retrieval and machine learning communities to realize object-oriented versions that accommodate inexact matching over structured hypotheses with weighted attributes. We also exploit the object-oriented precision and recall metrics in additional metrics that account for the costs of false-positive and false-negative threat reporting.

We have reported on our scoring methods more fully previously; the present brief presentation is offered to help make this work accessible to the machine learning community.

Introduction

Information fusion—collecting individual, disparate items of information into coherent, structured reports to provide a holistic situation assessment—is qualitatively similar to machine learning classification in some ways and different in others. Both tasks produce hypotheses whose veracity may be tested against an available standard for a given problem instance. In information fusion, the standard, known as "ground truth," may be developed as part of a live or simulation-based experimental process. In supervised learning, the standard comes from class labels associated with training instances.

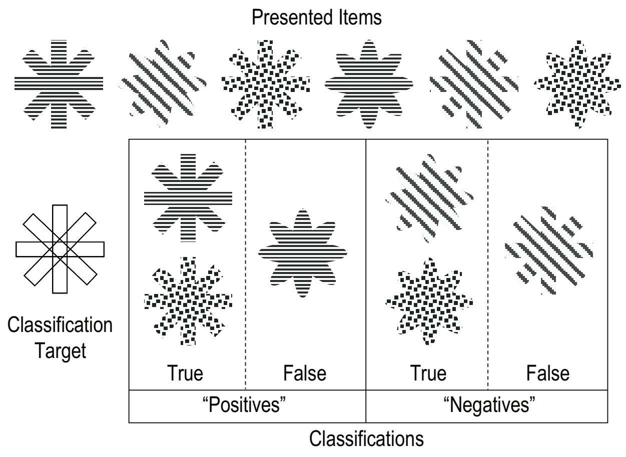


Figure 1: The binary classification task

In binary classification (deciding which instances are and which are not members of a target class), the standard partitions hypotheses crisply into true positives, true negatives, false positives, and false negatives, from which we may compute precision, recall, accuracy, and other metrics of interest. This is schematized in Figure 1. (Imagine that the visual cues are available to us as readers but that the technology under test must rely on contextual cues, not shown.)

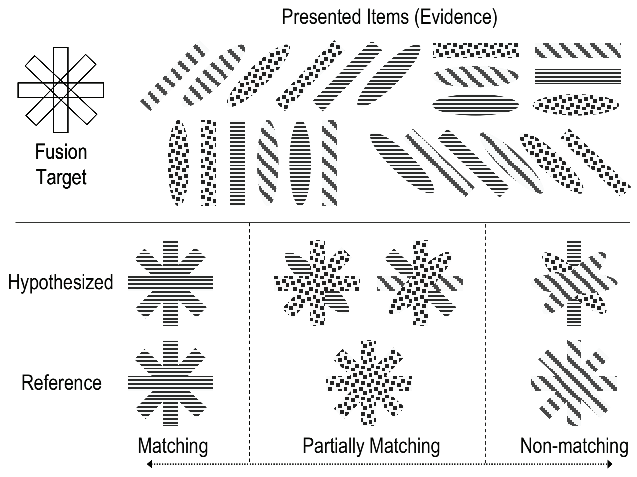


Figure 2: The information fusion task

In information fusion (even if there is only one target class), there are many possible combinations of existing information items into hypotheses that are structured as object instances with attribute values. We want to reward hypotheses that may not match a given ground truth instance exactly but are close. Approximate matching gives rise to a sort of continuum where the truth and falsity of being a match are not necessarily crisp as in binary classification. This is illustrated in Figure 2. (Imagine that the simple shapes falling along the four different axes correspond to outputs from four different sensors.) Along this continuum fall combinatorially many (possible) hypothesized instances that may defy the practical enumeration assumed by machine learning's accuracy metric. In determining how closely two instances match each other, we sometimes need to accord different levels of importance or weight to their various attributes. For all these reasons, computing information fusion metrics demands a qualitatively different approach.

We have developed methods for scoring information fusion hypotheses that generalize traditional information retrieval metrics (precision, recall, F-value, area under the precision-recall curve). As we describe more fully below, given hypothesized instances (or "cases") and potentially matching instances from the standard (or "reference") for a given object class, we compute object-oriented precision and recall at two levels:

- Between pairs of hypothesized and reference cases ("case comparison");
- For the class as a whole, considering the best possible one-to-one (generally asymmetric) assignment of hypothesized to reference cases ("case pairing").

We apply this process recursively when attribute values are themselves (nested) object instances.

The methods are implemented in a performance evaluation laboratory for threat detection technologies, summarized by Schrag (2006). We have applied the overall laboratory approach including the present scoring methods in two distinct situation assessment domains (requiring only minor changes in the scoring implementation): counter-terrorism threat detection over an abstract, artificial world; and computer network intrusion detection. To illustrate scoring methods, we appeal here to the counter-terrorism domain, where the threat detection component is assumed to employ a variety of information fusion technology known as "link discovery" (LD) and is referred to here as an "LD" component.

The next section summarizes the counter-terrorism domain, including its case object classes and the connection of their instances in a directed acyclic graph facilitating computation of the metrics when object instances are nested. The section after that explains our scoring methods more fully. This brief presentation is oriented towards the machine learning community. More detail on the scoring methods and a discussion of related work are included in a longer paper (Schrag and Takikawa 2006) oriented towards the threat detection community. We appreciate that the 2007 AAAI Workshop on Evaluation Methods for Machine Learning II has invited resubmissions of relevant work. The present abstract and introduction are new, as are portions marked ***[For the machine learning community:]."*** Another paper (Schrag 2006) includes limited empirical results and analyses.

Counter-terrorism Threat Domain

Figure 3 exhibits some real-world motivation behind the abstract, artificial world challenge problem domain we have developed. On the left-hand side of Figure 3, "Farmer Fred" buys fertilizer and fuel oil and transports these *via* truck to his farm. He applies the fertilizer using his tractor which (along with his truck) burns the fuel oil. (Fred is an honest, hard-working man.) On the right-hand side, "Demolition Dan" acquires the same resources but mixes them into a slurry that he transports (*via* rental truck) to the basement of an office building. (Dan is up to no good.)

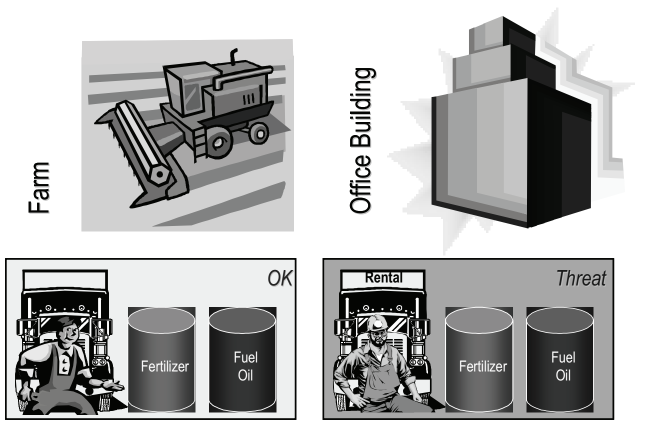


Figure 3: Real-world motivation for challenge problem

In the artificial world, capabilities (like farming and demolition) and resources (like fertilizer and fuel oil) are mapped to abstract elements that individuals can possess intrinsically or acquire. Infrastructure elements (like office buildings) are mapped to "targets" that support both legitimate/productive and destructive modes of use or "exploitation." Non-threat and threat individuals (like Fred and Dan) each may belong to any of various groups whose members collaborate in sub-group teams towards different goals.

Scoring-relevant type	Scoring-relevant attribute	Attribute domain	Reference attribute Weight	Reference attribute Cardinality
VulnerabilityExploitationCase				
	startingDate	Date (integer)	1	1
	endingDate	Date (integer)	2	1
	minAssetApplicationEndingDate	Date (integer)	2	1
	maxAssetApplicationEndingDate	Date (integer)	2	1
	performedBy	ThreatGroup	3	1
	directingAgent	ThreatIndividualEC	2	1
	deliberateActors	ThreatIndividualEC	1	1+
	targetInExploitation	ExploitationTarget	5	1
	modeInExploitation	VulnerabilityMode	4	1
ThreatGroup				
	exploitsVulnerabilities	VulnerabilityMode	1	1+
	memberAgents	ThreatIndividualEC	1	1+
ThreatIndividualEC				
	hasMember	ThreatIndividual	1	1+
ThreatIndividual				
ExploitationTarget				
VulnerabilityMode				
	modeCapabilities	Capability		1+
	modeResourceTypes	ResourceType		1+
Capability				
ResourceType				

Table 1: Scoring-relevant types and attributes

The challenge to LD is to hypothesize and report threat *cases*—top-level objects with attributes and values summarizing threat phenomena. We require LD to return hypothesis objects that are definite (incorporate neither logical nor probabilistic uncertainty) and that are structured according to a specification that summarizes key, relevant information (so that, *e.g.,* not every subevent of a hypothesized threat event need be detailed or considered during scoring). Table 1 presents the types and attributes that are considered during scoring, per the artificial world's representation. Along with each attribute is specified its domain, scoring weight, and reference attribute cardinality (either single or multiple).

The first four types in Table 1 are the scored case types. The remaining types are those that appear as scored attributes of cases or in turn as values of their attributes.

For each type, each instance also has a unique identifier (UID) by which it may be referred to. An object of class ThreatIndividualEC is used to represent an equivalence class (EC) of threat individual identities, supporting aliases. In attribute values, we interpret any of an EC's member UIDs as denoting the full EC.

Note that event objects have as attribute values objects of other types, some of which also are scored. We rely on the fact that our counter-terrorism domain's supercase type-to-subcase type graph (depicted in Figure 4) is directed and acyclic, as we compute scores for leaf types first and work our way up to root types. (In Figure 4, only the object types requiring case pairing are shown.)

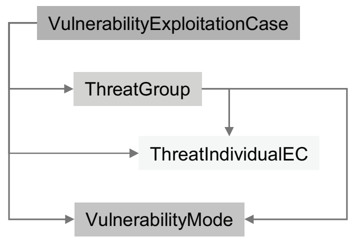


Figure 4: Counter-terrorism domain supercase type-to-subcase type graph

Hypothesis Scoring Methods

Figure 5 depicts the generic scoring scheme.

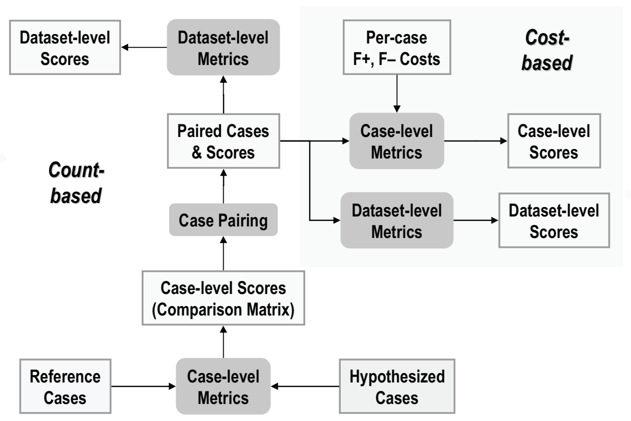


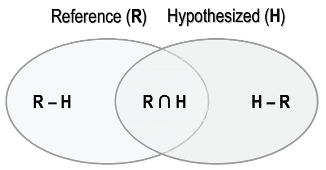
Figure 5: Generic scoring scheme

Reference cases are from ground truth, hypothesized cases from LD. We first compute scores for count-based metrics (precision, recall, and F-value), then for cost-based metrics.

Case objects have significant structure, and we want to credit LD when hypothesized cases approximately match reference cases. Match quality is determined by case comparison. Our dataset-level metrics associate each hypothesized case with no more than one reference case (and *vice versa*). When a hypothesized case object's existence has been inferred from lower-level evidence, we can decide which reference case to pair the hypothesized one with only by comparing the hypothesized with all relevant reference cases—on the basis of their attribute values. We store comparison results for the candidate pairs in a matrix. With inexact matching, it also can be ambiguous which of the one-to-one mappings admitted by the candidate pairs should be selected, so we use an algorithm that optimizes dataset-level scores. Using scores for count-based metrics and specified per-case costs of false-positive (F+) and false-negative (F–) reporting, we additionally compute scores for cost-based metrics.

Case Comparison and Pairing

Case comparison determines the quality of match between any two cases. We characterize this quality by specializing the traditional precision and recall metrics (which presume exact matching between hypothesized and reference items) illustrated in Figure 6.



$$\text{Recall } (R) = \frac{| R \cap H |}{| R - H \cup R \cap H |}$$

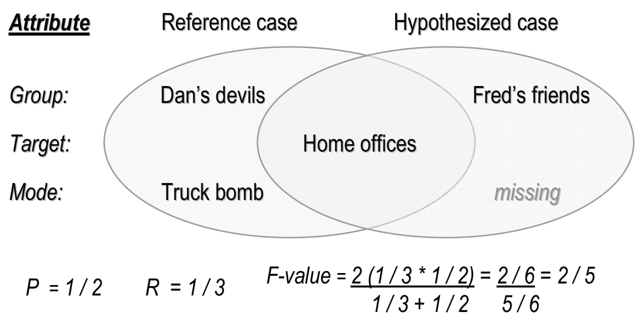
$$\text{Precision } (P) = \frac{| R \cap H |}{| R \cap H \cup H - R |}$$

Figure 6: Traditional precision and recall

Traditionally, recall R is the number of valid hypotheses divided by the number of detection targets (the required number of valid hypotheses). Precision P is the number of valid hypotheses divided by the number of all hypotheses.

Because a single metric to summarize the values of recall and precision is frequently useful, traditionally an appeal is made to F-value = $2PR / (P + R)$—the harmonic mean of precision and recall. (When both precision and recall are zero, we define F-value as zero.) F-value (also known as "F-score" or "F-measure") has the same extremes as a simple average of precision and recall but discounts differences between them more steeply (however not as steeply as min(P, R)).

To accommodate inexact matching over structured case objects, we define object-oriented versions of precision, recall, and F-value, as illustrated in Figure 7. (Our complete definitions—in the next section—address object attributes that may be weighted differently, so that attributes contribute to scores non-uniformly. See Figure 11.) ***[For the machine learning community:*** When comparing individual cases, we appeal to recall and precision—rather than accuracy—to accommodate object classes admitting attributes with arbitrarily many values (illustrated in Figure 12) that render impractical the enumeration of "true" negatives (or potential false negatives) used in computing accuracy. (In Figure 12, a hypothesized threat group may include arbitrarily many false-positive members or modes.) Note that in the fusion setting it would fall to the scoring method to perform this enumeration, as the set of all possible instances admitted by the evidence or by the supporting ontology is not normally materialized. ***]***



$$P = 1/2 \qquad R = 1/3 \qquad \text{F-value} = \frac{2\,(1/3 * 1/2)}{1/3 + 1/2} = \frac{2/6}{5/6} = 2/5$$

Figure 7: Object-oriented count-based metrics

Of the three attribute values in the reference case of Figure 7, the hypothesized case agrees only for the "Target" attribute, so the object-oriented recall score *R* is 1/3. Of the two attributes included in the hypothesis, only one agrees with the reference, so the object-oriented precision score *P* is 1/2. The corresponding object-oriented F-value is 2/5, as shown.

Case pairing determines which hypothesized cases to pair with which reference cases—since this may not be obvious, as illustrated in Figure 8.

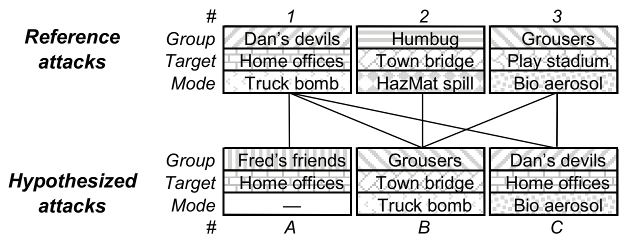
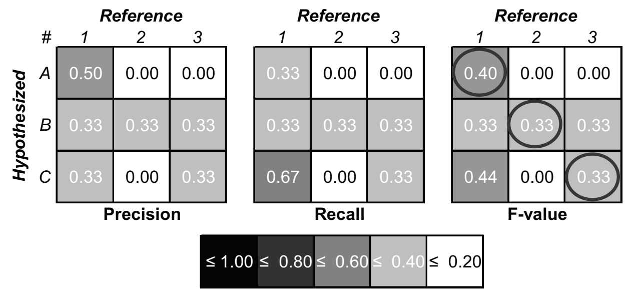


Figure 8: Case pairing issue

In Figure 8, we have three reference and three hypothesized attack cases. (Reference Case 1 and Hypothesized Case A correspond to the pairing of Figure 7.) Links appear in the bipartite graph between reference and hypothesized cases wherever these share one or more attributes. Figure 9 illustrates how we perform one-to-one case pairing using a matrix over all possible pairings. We forcibly pair any hypothesized and reference objects that have the same UID, and we omit them from such a case pairing matrix. When the numbers of reference and hypothesized cases do not match, we (effectively) pad the matrix, as necessary to make it square, with null cases. Precision and recall with respect to null cases are defined as zero.



Precision

#	1	2	3
A	0.50	0.00	0.00
B	0.33	0.33	0.33
C	0.33	0.00	0.33

Recall

#	1	2	3
A	0.33	0.00	0.00
B	0.33	0.33	0.33
C	0.67	0.00	0.33

F-value

#	1	2	3
A	0.40	0.00	0.00
B	0.33	0.33	0.33
C	0.44	0.00	0.33

Reference
Hypothesized
≤ 1.00
≤ 0.80
≤ 0.60
≤ 0.40
≤ 0.20

Figure 9: Case pairing matrix and metrics

In Figure 9, we compute per-pair object-oriented precision, recall, and F-value (as in Figure 7). Then we use an optimization algorithm to select (red-circled) pairs leading to the greatest average object-oriented F-value. (So, we have computed a matching for Figure 8's bipartite case graph including just the strictly vertical edges.)

Generally, we admit entries to candidacy per a user-specified threshold reflective of hypothesis F-value deemed adequate for an analyst or other consumer. ***[For the machine learning community:*** Note that although thresholding crisply (if perhaps artificially) partitions hypotheses into true positives and submitted false negatives, it does not in so doing materialize the full set of conceivable false negatives that would be necessary to compute machine learning's accuracy metric. ***]***

We also can specify additional necessary conditions (besides the threshold) for a candidate pair's inclusion in the scoring matrix. *E.g.,* we require that hypothesized and reference threat event cases temporally overlap.

When LD can rank its hypotheses with respect to estimated quality, this ranking supports developing a precision-recall curve and computing the area under the curve (AUC)—as illustrated in Figure 10. Any consistently applied variant of precision and recall—*e.g.,* using any consistent F-value threshold—suffices here.

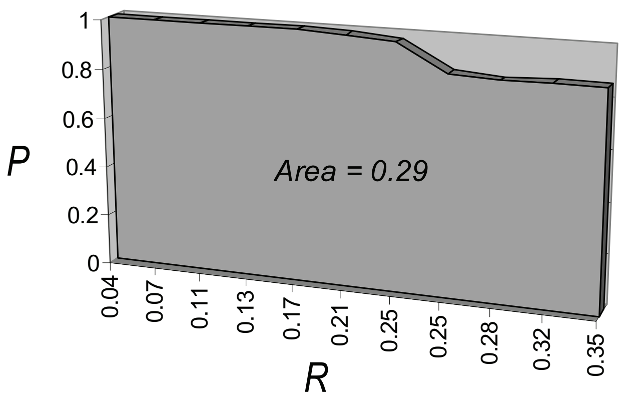


Figure 10: Precision-recall curve and area

Considering LD's highest-ranked hypotheses first, at each successive point, we compute precision and recall (with respect to the full reference case set) over the partial hypothesis set including all the previous hypotheses plus LD's next-ranked hypothesis. Instead of performing full case pairing at each point, we expediently take the case pairings over the full sets of reference and hypothesized cases as authoritative and impose them as we consider each successively presented case to develop the curve.

Note that our detection task, where structured threat hypotheses must be developed from lower-level evidence, constrains the precision-recall curve less than the traditional information retrieval task, where every presented item merely must be classified as either positive or negative. In the traditional setting, the minimum expected AUC is 0.5. In our setting, some reference threats may never be reported, given whatever practical minimum-estimated precision threshold a detector may set. (The universe of potential—*e.g.,* syntactically admissible—hypotheses for a given dataset is practically unbounded.) Accounting also for inexact matching, neither of object-oriented precision and recall need take the curve to its maximum value of one. So, in our setting, the range of AUC range is [0, 1].

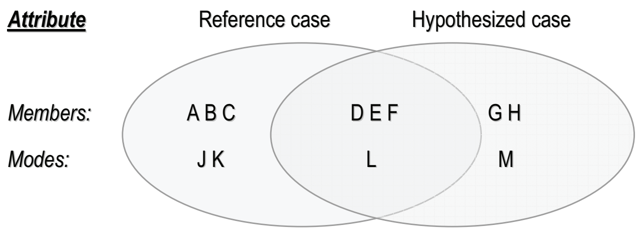
[For the machine learning community: The size of the information fusion hypothesis space also presents issues for some other metrics commonly used in performance analysis of (binary) classifiers. In particular, the so-called "false-positive rate" used in receiver operator characteristic (ROC) curves and in the calculation of the "detection capability"[1] metric of Gu *et al.* (2006) assumes a practically enumerated set of "true-negative" responses (*i.e.,* the presented items known in ground truth to be non-threat). True negatives also must be enumerated for the machine learning community's commonly used "accuracy" metric (the number of true—positive and negative—responses divided by the number of all—true and false—responses). ***]***

Case Comparison Detail

Here we describe how to compare two like-type cases to determine their object-oriented precision P and object-oriented recall R.

We treat a case as a set of assertions regarding the case's attribute values—*e.g.,* (hasMembers Group-4931 Individual-2437). Note that a given case can have multiple, distinct assertions pertaining to a given (multi-valued) attribute. *E.g.,* Group-4931 can have more than one member. Note that the reference and hypothesized cases can have different numbers of assertions and of attributes, depending on numbers of values per attribute reported by the reference and by the hypothesis. This is illustrated for a pair of ThreatGroup cases in Figure 11.

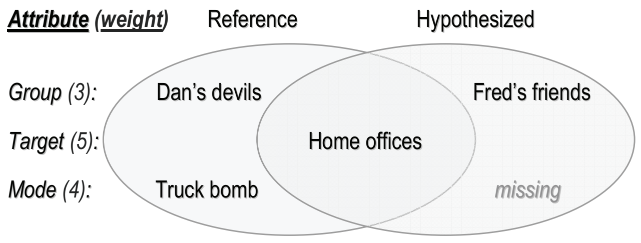
[1] The authors, working in the domain of computer network intrusion detection, describe an "intrusion detection capability" metric which is in fact applicable in any binary classification setting.



$P = 4/7 \quad R = 4/9 \quad F\text{-value} = \dfrac{2\,(4/7 * 4/9)}{4/7 + 4/9} = \dfrac{32/63}{64/63} = 1/2$

Figure 11: Case comparison with multi-valued attributes

For each case type, for each defined attribute, an assertion weight is specified in a case scoring specification as summarized in Table 1. For a given attribute, the same weights are used for assertions of hypothesized as for those of reference cases. Figure 12 adds non-uniform attribute weights to the example of Figure 7. The metrics' sensitivities to specified attribute weights depends on the attributes' relative cardinality (how many times each weight is counted) and—for nested objects—on weights applied in supercases.



$R_b = 3 + 5 + 4 = 12$
$P_b = 3 + 5 = 8$
$R_c = P_c = 5$

$R = R_c / R_b = 5/12$
$P = P_c / P_b = 5/8$
$F\text{-value} = \dfrac{2\,(5/12 * 5/8)}{5/12 + 5/8} = \dfrac{50/96}{100/96} = 1/2$

Figure 12: Case comparison with attribute weights

For a given reference case with the set of assertions $\{r_1, r_2, \ldots, r_m\}$ and corresponding set of weights $\{w_1, w_2, \ldots, w_m\}$, we define the "object-oriented recall basis" $R_b = \sum_{(i=1\ldots m)} w_i$. (So, each weight is counted once for each assertion in which the attribute appears.) For a given hypothesized case with the set of assertions $\{h_1, h_2, \ldots, h_n\}$ and corresponding set of weights $\{w_1, w_2, \ldots, w_n\}$, we similarly define the "object-oriented precision basis" $P_b = \sum_{(j=1\ldots n)} w_j$. Note that, for a given comparison of two cases, R_b and P_b may differ depending on numbers of values per attribute reported by the reference and by the hypothesis.

We pair reference and hypothesis attribute assertions one-to-one, computing for each pair (r_i, h_i) the following (per the next section's rules for assertion comparison).

- Object-oriented recall $R(r_i, h_i)$
- Object-oriented precision $P(h_j, r_j)$

We define the "object-oriented recall contribution" R_c as the sum over the hypothesized case's assertions of assertion weight w_i discounted by corresponding recall—$R_c = \sum_{(i=1\ldots n)} R(r_i, h_i) * w_i$. The "object-oriented precision contribution" P_c is the sum over the reference case's assertions of assertion weight w_j discounted by corresponding precision—$P_c = \sum_{(j=1\ldots m)} P(h_j, r_j) * w_j$.

For a given pair of reference and hypothesized cases, we have the following.

$R = R_c / R_b$
$P = P_c / P_b$
$F\text{-}value = 2 (P * R) / (P + R)$

We compute the metrics for a given dataset's cases of a given type as follows. Let N_R be the number of reference cases and N_H the number of hypothesized cases. Let the set $\{p_1, p_2, \ldots, p_o\}$ be the computed pairs, and $R(p_k)$, $P(p_k)$ the object-oriented recall and precision (respectively) of the kth pair. Then for the dataset we have the following.

$R = (\sum_{(k=1\ldots o)} R (p_k)) / N_R$
$P = (\sum_{(k=1\ldots o)} P (p_k)) / N_H$

Pairing and Comparison for Attribute Values

To compare a given hypothesized case with a given reference case, we require a one-to-one pairing among their respective attribute assertions. This pairing requires comparison of attributes values, , which may be either objects or scalars (*e.g.,* Dates). We define the object-oriented precision and recall for a pair of reference and hypothesized Dates using a unit-normalized temporal distance metric. We require that paired assertions have the same attribute, so single-valued attributes pair straightforwardly and multi-valued attributes require a one-to-one pairing over their values. We use a global one-to-one pairing between all the hypothesized and reference instances of the nested case types for any multi-valued attributes. Our directed acyclic supercase type-to-subcase type graph (Figure 4) enables us to pre-compute this pairing. Once pairs have been established, we can read off each pair's object-oriented precision and recall.

Cost-based Metrics

Costs are associated (*via* scoring specifications) with F+ and F– reports of a case, indicating (respectively) the costs of actions taken in response to the false report of a threat and of inaction resulting from a threat's non-report. We define the "assessed cost" metric to discount these costs based on a hypothesized case's quality, as follows.

- To the extent (corresponding to the factor (1 – *P*)) that the hypothesis includes content beyond that of its paired reference case, we assess the associated F+ cost.
- To the extent (corresponding to the factor (1 – *R*)) that the hypothesis lacks content included in its paired reference case, we assess the associated F– cost.

Dataset-level metrics sum over case-level metrics for the selected pairs. For consistency with the scale of the count-based metrics, we use the "normalized assessed cost" (NAC), dividing the assessed cost for a dataset by the sum of the F– costs for the reference cases (*i.e.,* normalizing by the score that would result if no hypotheses were returned). Note that the normalized assessed cost can be greater than one, as there is no inherent limit on the number of F+ hypotheses. For consistency with the quality polarity of the count-based metrics (*i.e.,* one being better than zero), we use the unit complement of NAC (1 – NAC), or "normalized assessed cost complement" (NACC).

Cost-based metrics may be appealed to (rather than F-value) to provide an alternative, linear formulation of hypotheses' utility. Their status could be promoted by using one of these metrics, rather than F-value, in the case comparison matrices supporting case pairing.

Conclusion

These methods are applicable in principle to performance evaluation in any domain where technologies under test return instances of one or more structured object types, given a problem for which a scoring standard or answer key is available.

Acknowledgements

We thank our colleagues who have contributed to the hypothesis scoring design, notably (with apologies to anyone we overlook) Chris Boner, Hans Chalupsky, Thayne Coffman, Paul Goger, Ian Harrison, David Jensen, Kathy Laskey, Andrew Moore, Peter Olson, Josh Powers, Afzal Upal, Andre Valente, and Ed Wright. We additionally thank Randy Dieterle, Brandon Goldfedder, Tod Levitt, and anonymous reviewers for their comments.

References

Gu, G., Fogla, P., Dagon, D., Lee, W., and Skoric, B. 2006. Measuring Intrusion Detection Capability: An Information-Theoretic Approach, *Proceedings of ACM Symposium on InformAtion, Computer and Communications Security (ASIACCS'06).*

Schrag, R. 2006. A Performance Evaluation Laboratory for Automated Threat Detection Technologies. *Performance Metrics for Intelligent Systems Workshop.* National Institute of Standards and Technology.

Schrag, R. and Takikawa, M. 2006. Scoring Hypotheses from Threat Detection Technologies. *AAAI Fall Symposium on Capturing and Using Patterns for Evidence Detection.*