Slide 1

Incremental Learning with Partial Instance Memory

Mark Maloof

Department of Computer Science
Georgetown University
Washington, DC

maloof@cs.georgetown.edu
http://www.cs.georgetown.edu/~maloof

Joint work with:
Ryszard Michalski (GMU & Polish Academy of Science)

XIII International Symposium on Methodologies for Intelligent Systems
Lyon, France

27 June 2002

Slide 2

Talk Overview

- On-line learning, concept memory, instance memory
- Algorithm for learning with partial instance memory
- Selecting *extreme examples*, those on the boundaries of concept descriptions
- Evaluation and Comparison
  - computer intrusion detection
  - the STAGGER Concepts
- Future Work

Slide 3

On-line Learning

- Training examples distributed over time
- But system must always be able to perform
- Temporal-Batch Learning
  1. Learn, say, rules from examples
  2. Store rules, store examples
  3. Use rules to predict, navigate, etc.
  4. When new examples arrive, add to current examples
  5. Goto step 1
- Incremental Learning
  1. Learn rules from examples
  2. Store rules, discard examples
  3. Use rules to predict, navigate, etc.
  4. When new examples arrive, learn new rules using old rules and new instances
  5. Goto step 2

Slide 4

Concept Memory

- Full: Learner stores concept descriptions, changing them only when new examples arrive (e.g., WINNOW)
- No: Learner stores no concept descriptions that generalize training examples (e.g., IB2)
- Partial: Learner stores concept descriptions and modifies them but not necessarily in response to the arrival of new training examples, like weight decay (e.g., FAVORIT)

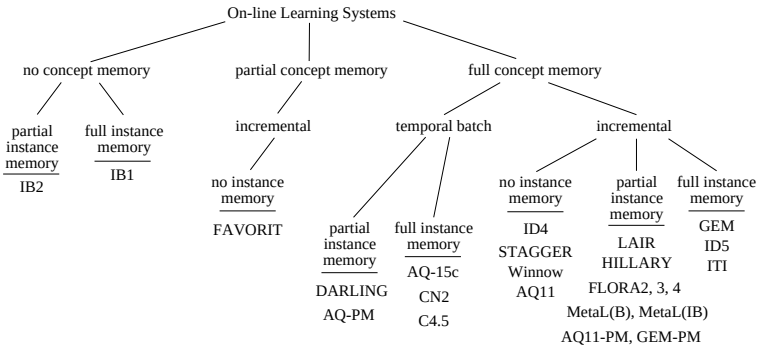
Slide 5

Instance Memory

- Full: Learner stores all examples from the input stream (e.g., ID5, GEM)
- No: Learner stores no examples (e.g., ID4, AQ11)
- Partial: Learner stores *some* examples (e.g., LAIR, HILLARY, FLORA, DARLING, METAL(B), METAL(IB), AQ-PM, AQ11-PM)

Slide 6

Classification of Learning Systems



Slide 7

Algorithm for Learning with Partial Memory

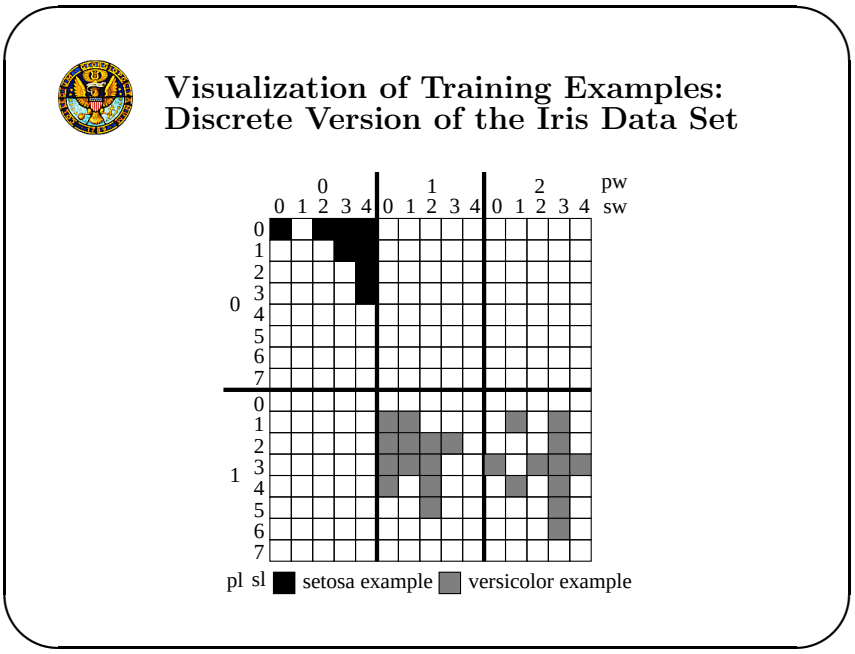
1. Learn rules from training examples
2. Select a portion of the examples
3. Store rules, store examples
4. Use rules to predict, navigate, etc.
5. When new examples arrive
   - learn new rules using
     - old rules,
     - new instances, and
     - examples held in partial memory
6. Combine new instances with those in partial memory
7. Goto step 2

Slide 8

Selecting Examples for Partial Memory

- LAIR: the first positive example only
- HILLARY: only the negative examples
- DARLING: examples near the centers of clusters
- IB2: misclassified examples
- METAL(B), METAL(IB): sequence over a fixed window of time
- FLORA: sequence over a changing window, set adaptively
- AQ-PM, AQ11-PM, GEM-PM: examples on the boundaries of rules (i.e., *extreme examples*), possibly over a fixed window of time

Slide 9



Visualization of Training Examples: Discrete Version of the Iris Data Set

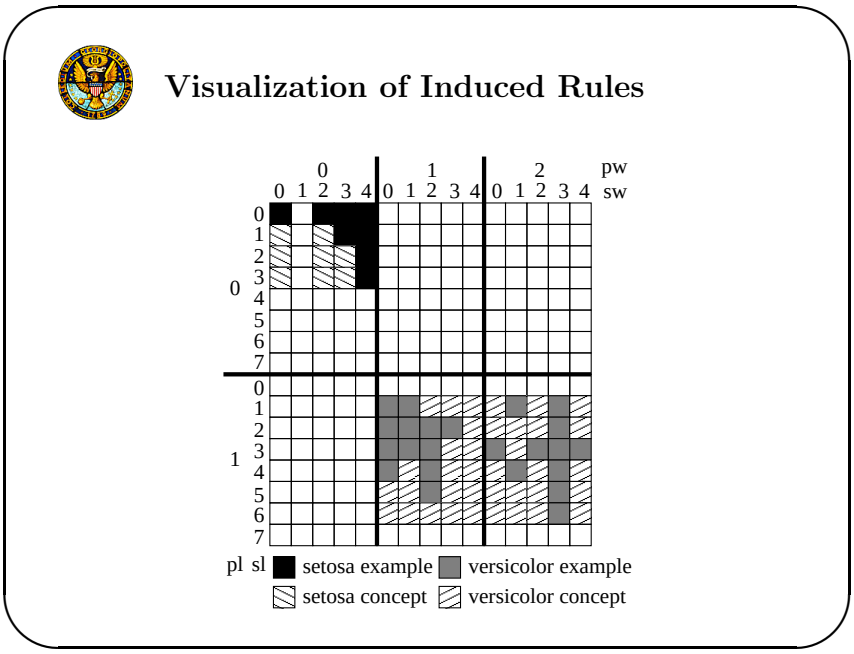
Slide 10

Induced Characteristic Rules

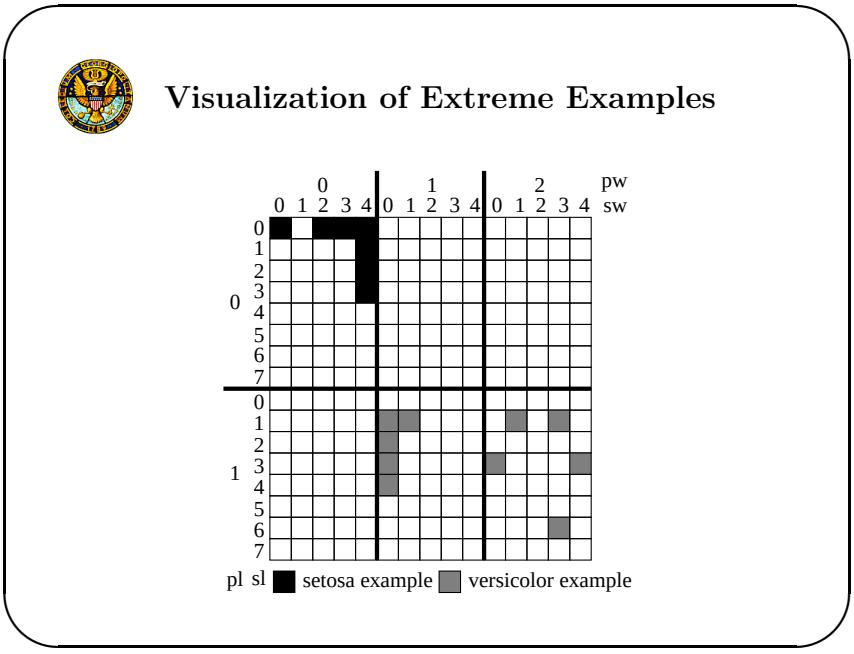
```
setosa ← [pl = 0] & [pw = 0] &
         [sl = 0..3] & [sw = 0, 2..4]

versicolor ← [pl = 1] & [pw = 1..2] &
             [sl = 1..6] & [sw = 0..4]
```

Slide 11



Slide 12



Slide 13

Evaluation of Learning Systems

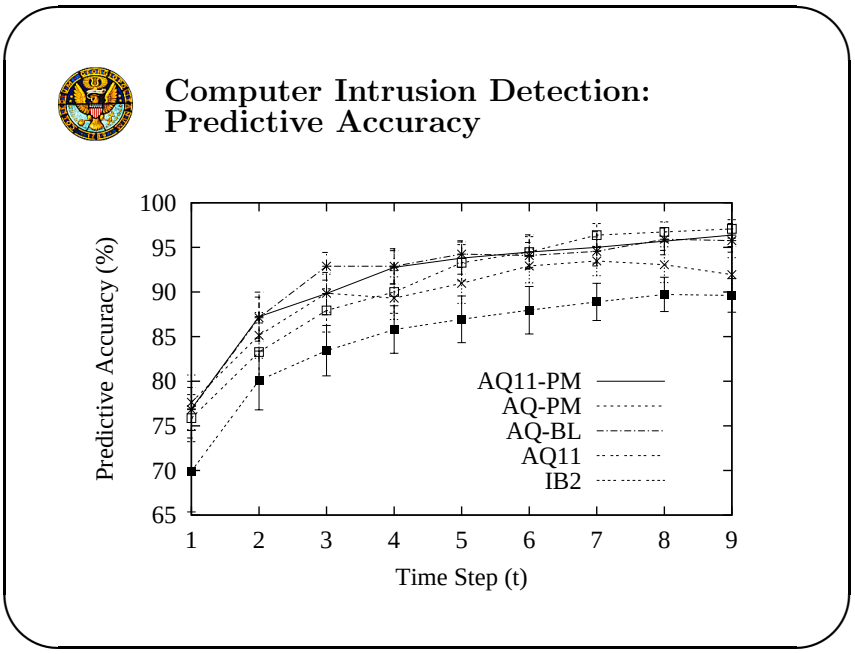
- AQ11-PM: Incrementally learns disjunctive rules. Selects examples on the boundaries of these descriptions over a fixed window of time.
- AQ11: Incrementally learns disjunctive rules. No instance memory. A lesioned version of AQ11-PM. Pascal implementation
- AQ-PM: Temporal-batch learner. Disjunctive rules. Selects examples on the boundaries of these descriptions over a fixed window of time. C implementation
- AQ-BL: Temporal-batch learner. Disjunctive rules. Full instance memory. A lesioned version of AQ-PM. C implementation
- FLORA2: Incrementally learns disjunctive rules. Selects examples over a window of time. Heuristic adjusts window size
- IB2: Instance-based learner. Selects misclassified examples

Slide 14

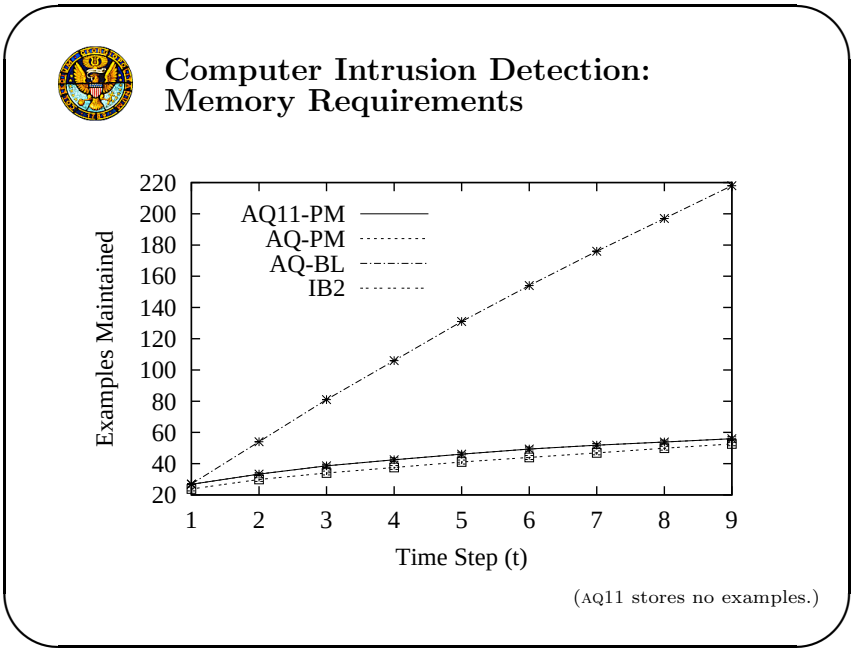
Computer Intrusion Detection

- Learning behavioral profiles of computing use for detecting intruders (also misuse)
- Derived our data set from the UNIX acctcom command
- Three weeks, over 11,200 records, selected 9 of 32 users
- Segmented into *sessions*: logouts and 20 minutes of idle time
- For each session, computed minimum, average, and maximum for seven numeric metrics
- Selected 10 most relevant: maximum real time, average and maximum system and user time, average and maximum characters transferred, average blocks read and written, maximum CPU factor, average hog factor
- Divided data into 10 partitions, used 1 for testing, 9 for training, applied methods, and repeated 30 times

Slide 15

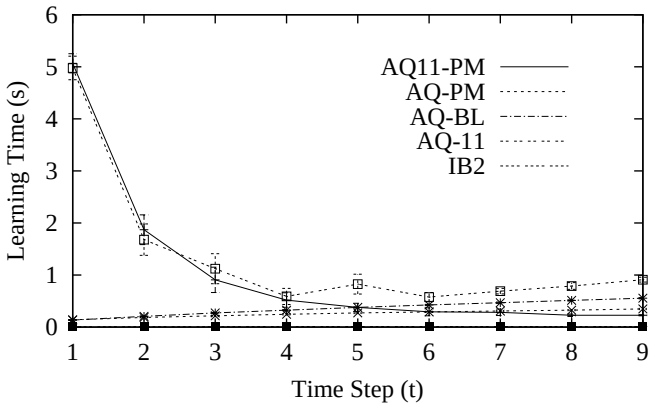


Slide 16



Slide 17

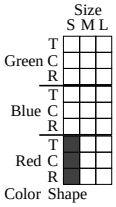
Computer Intrusion Detection: Learning Times



Slide 18

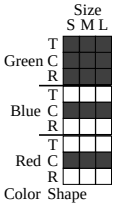
The STAGGER Concepts

`(size = small)` & `(color = red)`



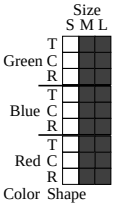
a. Target concept for time steps 1–39.

`(shape = circle)` ∨ `(color = green)`



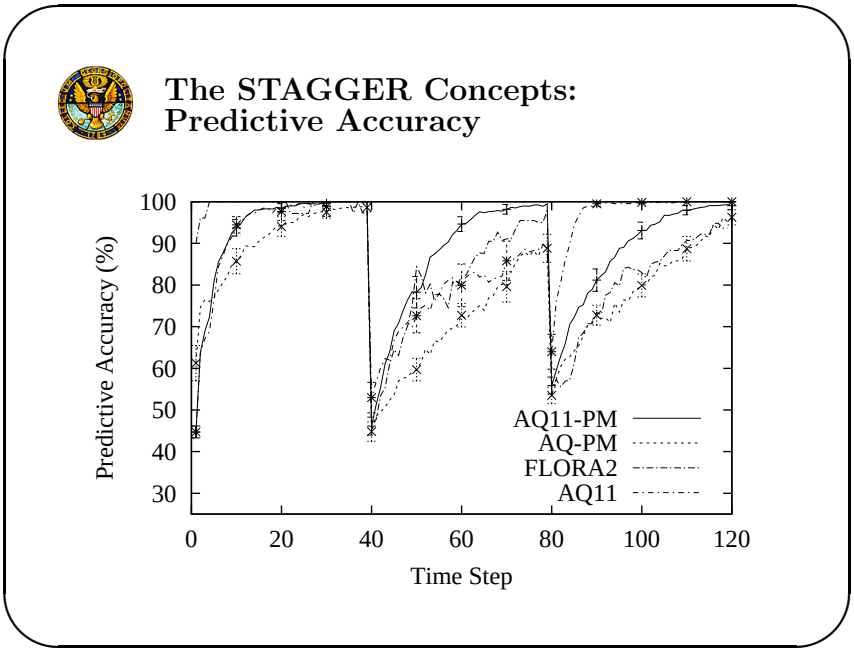
b. Target concept for time steps 40–79.

`(size = medium, large)`

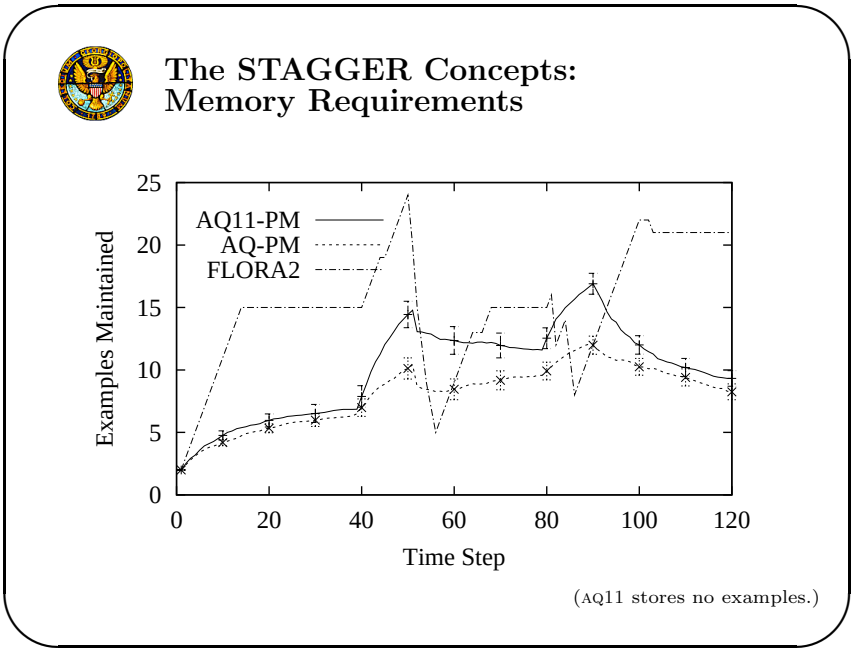


c. Target concept for time steps 80–120.

Slide 19



Slide 20



Slide 21

Observations

- For static concepts, partial-memory learners, as compared to lesioned versions, tend to:
  - decrease predictive accuracy—often slightly
  - decrease memory requirements—often significantly
  - decrease learning time—often significantly
  - can decrease concept complexity
  - has little effect on performance time
- For changing concepts,
  - track concepts better than incremental learners with no instance memory (e.g., STAGGER, AQ11)
  - AQ11-PM tracks concepts comparably to FLORA2

Slide 22

Future Work

- Better characterization of performance using synthetic data sets: CNF, DNF, m-of-n, class noise, concept overlap
- Scale to larger data sets: Acquired more audit data, $\sim$10 GB
- Evaluate effect of skewed data: Rooftop detection
- Prove bounds for predictive accuracy and examples maintained
- Heuristics to adapt size of forgetting window

References

Aha, D., Kibler, D., & Albert, M. (1991). Instance-based learning algorithms. *Machine Learning*, *6*, 37–66.

Elio, R., & Watanabe, L. (1991). An incremental deductive strategy for controlling constructive induction in learning from examples. *Machine Learning*, *7*, 7–44.

Iba, W., Woogulis, J., & Langley, P. (1988). Trading simplicity and coverage in incremental concept learning. In *Proceedings of the Fifth International Conference on Machine Learning* (pp. 73–79). San Francisco, CA: Morgan Kaufmann.

Kubat, M., & Krizakova, I. (1992). Forgetting and aging of knowledge in concept formation. *Applied Artificial Intelligence*, *6*, 195–206.

Littlestone, N. (1991). Redundant noisy attributes, attribute errors, and linear-threshold learning using Winnow. In *Proceedings of the Fourth Annual Workshop on Computational Learning Theory* (pp. 147–156). San Francisco, CA: Morgan Kaufmann.

Maloof, M. (1996). *Progressive partial memory learning.* Doctoral dissertation, School of Information Technology and Engineering, George Mason University, Fairfax, VA.

Maloof, M., & Michalski, R. (2000). Selecting examples for partial memory learning. *Machine Learning*, *41*, 27–52.

Michalski, R., & Larson, J. (1983). *Incremental generation of VL_1 hypotheses: The underlying methodology and the description of program AQ11* (Technical Report No. UIUCDCS-F-83-905). Department of Computer Science, University of Illinois, Urbana.

Reinke, R., & Michalski, R. (1988). Incremental learning of concept descriptions: A method and experimental results. In J. Hayes, D. Michie, & J. Richards (Eds.), *Machine intelligence 11* (pp. 263–288). Oxford: Clarendon Press.

Salganicoff, M. (1993). Density-adaptive learning and forgetting. In *Proceedings of the Tenth International Conference on Machine Learning* (pp. 276–283). San Francisco, CA: Morgan Kaufmann.

Schlimmer, J., & Fisher, D. (1986). A case study of incremental concept induction. In *Proceedings of the Fifth National Conference on Artificial Intelligence* (pp. 496–501). Menlo Park, CA: AAAI Press.

Utgoff, P. (1988). ID5: An incremental ID3. In *Proceedings of the Fifth International Conference on Machine Learning* (pp. 107–120). San Francisco, CA: Morgan Kaufmann.

Widmer, G. (1997). Tracking context changes through meta-learning. *Machine Learning*, *27*, 259–286.

Widmer, G., & Kubat, M. (1996). Learning in the presence of concept drift and hidden contexts. *Machine Learning*, *23*, 69–101.

