Using Active Learning in Intrusion Detection

Magnus Almgren, Erland Jonsson

Chalmers University of Technology

Objective

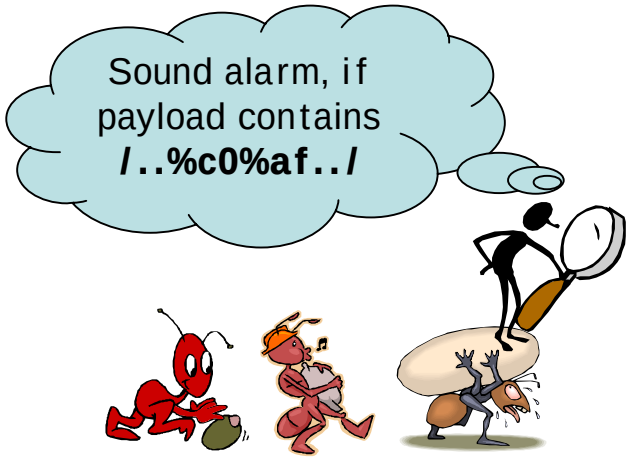
- Today's state of the art of intrusion detection:
  - An expert must *manually* write a rule to detect new attacks.
  - This process is
    - expensive, as the expert must be present,
    - slow, as a human is involved, and finally
    - inflexible, meaning that it is difficult to adapt the systems to local site conditions.
- ***Objective:***
  Reduce the amount of labeled data required for self-learning intrusion detection systems.

Outline

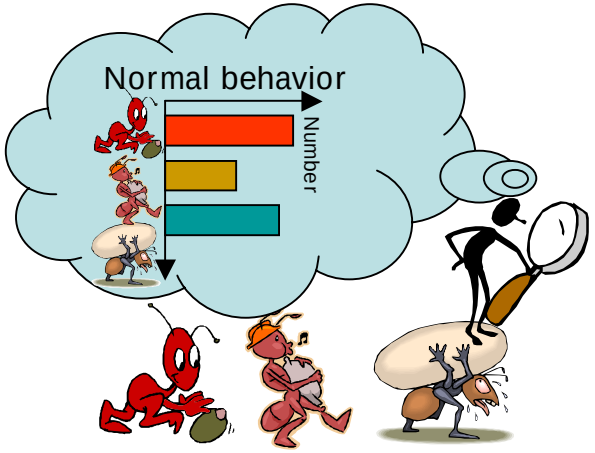
- Background
- Introduction to ***Active Learning***
- Experiment Details
- Results
- Conclusion

Classification of IDS: Detection Models

- Misuse detection (signature-based)
  - Define what is ***wrong*** and give alarms for such behavior (*default permit)*



- Anomaly Detection
  - Define what is **correct** and give alarms for everything else (*default deny)*



Detection Model Characteristics

- Misuse Detection
  - Defines malicious behavior
  - Experts design attack "signatures"
  - Few(er) false alarms
  - Restricted ability to generalize

- Anomaly Detection
  - Defines normal behavior (traditionally)
  - Self-learning algos used
  - Can generalize to new attacks, but
  - Plagued by a high false alarm rate

- Hybrid systems
  - Defines both models (in some way)
  - Significant advantages – "best of both worlds"
    - Automatic learning: cheap?, fast?, veracious?
    - Detects unknown (novel) attacks, and
    - abuse-of-privilege attacks (masqueraders, insider misuse)
  - Promising for future
  - Need extensive amount of (labeled) data

Data sets in Intrusion Detection (1)

- Authentic data sets are difficult to come by in intrusion detection
- Real data collected at live sites
  - No ground truth
  - Cannot be shared due to privacy issues
  - No comparison possible because different live sites different
  - Often assumed for anomaly: No attacks in this dataset
- Attack data collected from artificial experiments/environments
  - Red Team attack
    - Expensive, and often not released. Also, no ground truth and might not be representative
  - Capture The Flag (DefCon etc)
    - No ground truth, not representative of normal traffic

Data sets in Intrusion Detection (2)

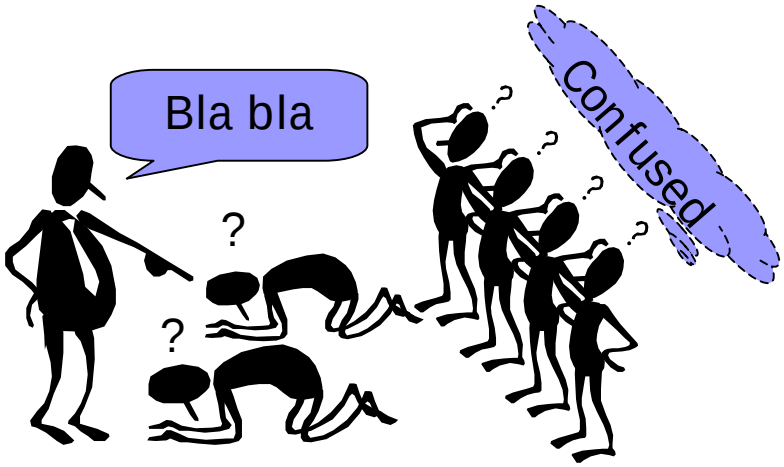
- Creating synthetic data (Lundin)
  - Need accurate seeding
  - Statistical distribution should be correct
  - Difficult to find variations on attacks
- Simulated data sets
  - For example: Lincoln Labs experiment
  - Expensive and difficult undertaking
  - Criticized by McHugh and others

Outline

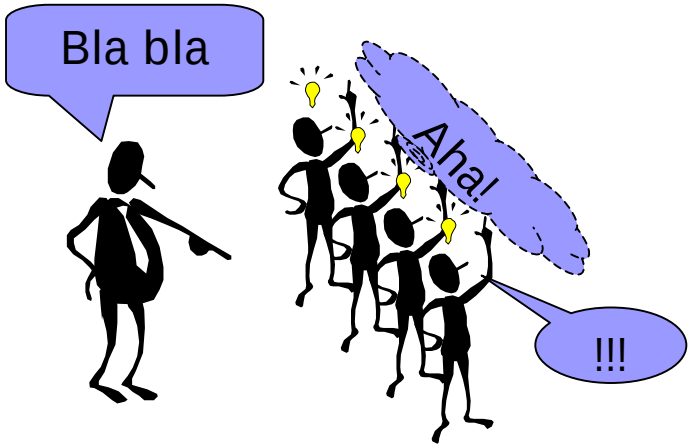
- Background
- Introduction to *Active Learning*
- Experiment Details
- Results
- Conclusion

Active Learning
Human Teaching Analogy

- Old Scheme
  - ***Off line*** labeling
  - Lecture w/one-way communication



- Active Learning
  - ***Interactive*** labeling
  - Lecture w/ Q&A



Active Learning: Basic Idea

- All Machine Learning algorithms need lots of data but data expensive to come by …
- **Past:** Throw random data at the algorithm
- **Now:** Algorithm ***actively*** chooses what type of data it wants to train upon
  - Active Learning
  - Query Learning
  - Uncertainty Sampling

Active Learning: Background

- Suggested during the ’80s but not used for IDS.
- Different flavors:
  - Let algorithm suggest new (artificial) instances that it would like to get labeled.
    - Problem: Many such instances have no meaning to experts.
  - Use a pool of unlabeled examples that algorithm can choose from.
    - All examples are “real.”
    - A pool must exist, and an expert must be able to label them iteratively.

Definitions

- Pool of unlabeled examples: P *(iid)*
  Training set: D
- The active learner, l, has two separate components: (f, q)
  - f is any type of *classifier (i.e. a machine learning algo)*
  - q is a *query function*
- *In each step f is trained on D, and then q chooses new examples from P to be added to D.*

Support Vector Machines

- In this study, we mainly used SVMs.
  - Simplified, SVMs can be seen as the fitting of a straight line to data in a plane to a higher-dimensional case.
- SVM classifier, nonlinear function: $f(\mathbf{x}) = \langle \mathbf{w} \cdot \phi(\mathbf{x}) \rangle + b$
- ϕ is a nonlinear mapping from the input space to the feature space.
- The vector **w** defines a hyper plane that separates benign from malicious events, where
- **w** is constructed from some of the input data, the so-called *support vectors.*
- SVMs avoid the two problems of dimensionality:
  - generalize well to unseen data and they are
  - efficient by avoiding explicit use of higher-order dimensional spaces.

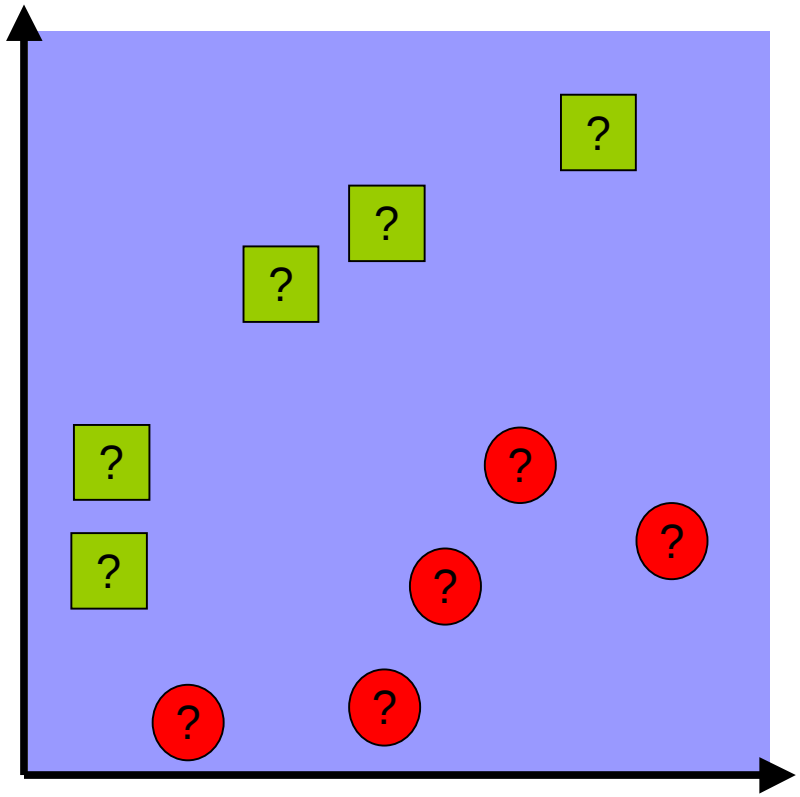
Query Functions

- *Several variants found in literature:*
  Predicted Loss, Voting Committees, etc.

- *Simple query function* *(taken from Tong et al.)*
  1. *Use the trained classifier on all examples in the pool P.*
  2. *Find the unlabeled example ($\boldsymbol{x}$,$\hat{y}$) closest to the decision boundary.*
  3. *Present the expert with this example to find the correct label and then add it to the training set D.*

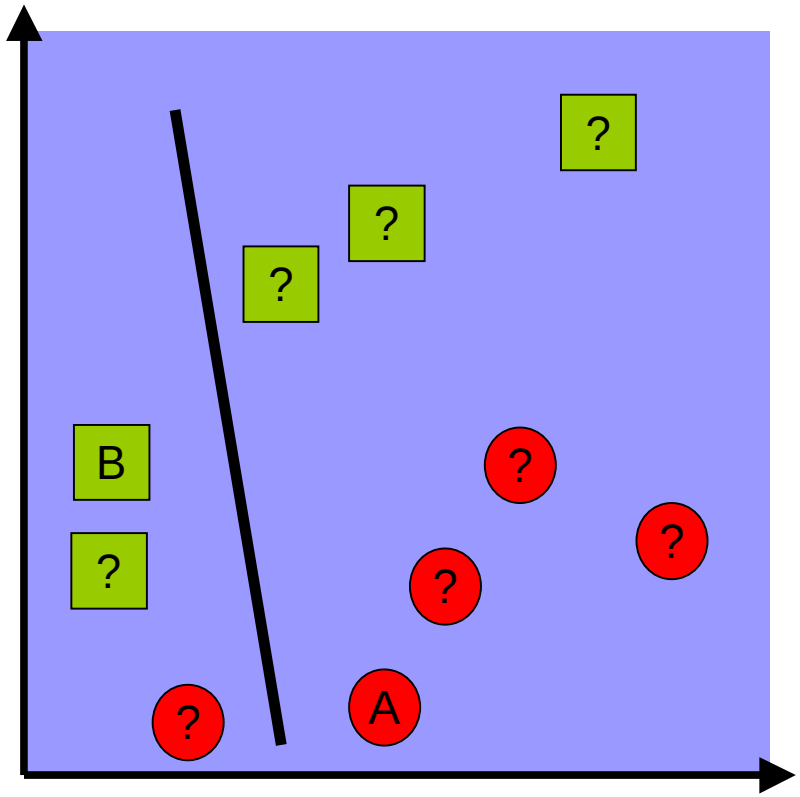
Active Learning Example

1. Take unlabeled pool of data



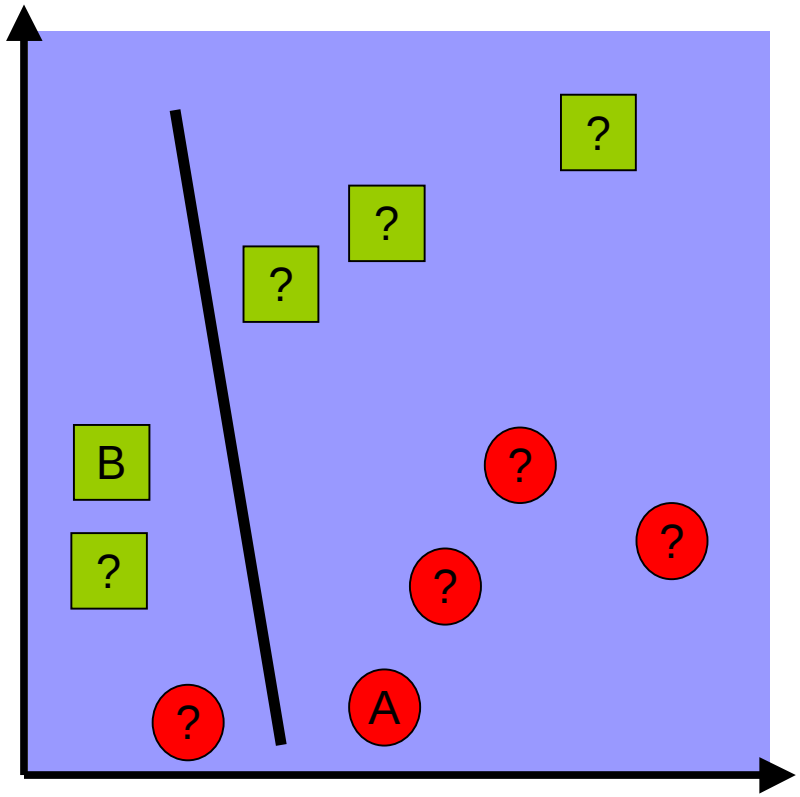
Active Learning Example

1. Take unlabeled pool of data
2. Label some data (seed)



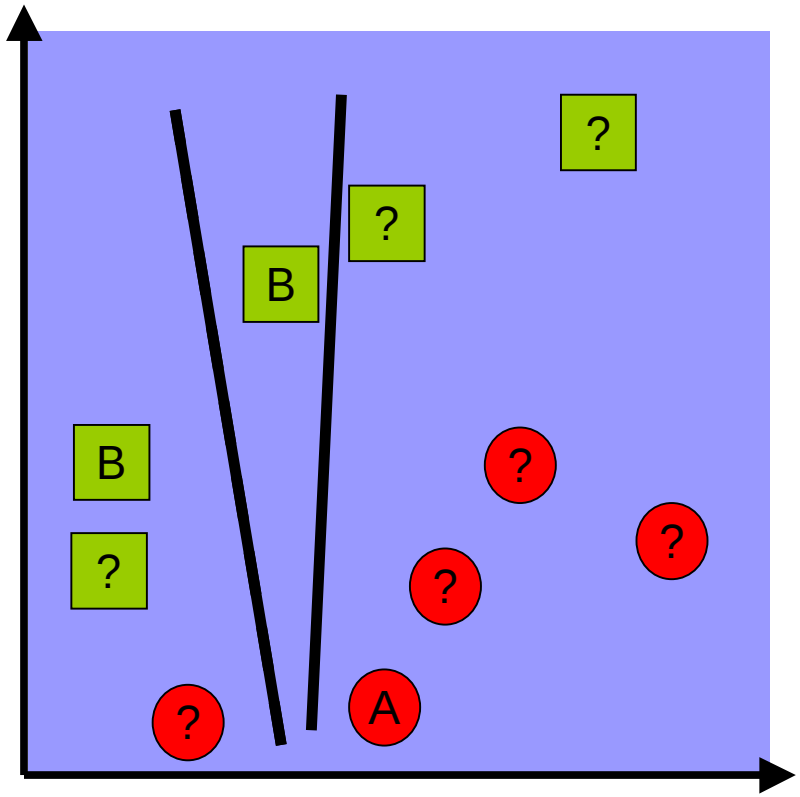
Active Learning Example

1. Take unlabeled pool of data
2. Label some data (seed)
3. Label each point of which the algo is ***uncertain***



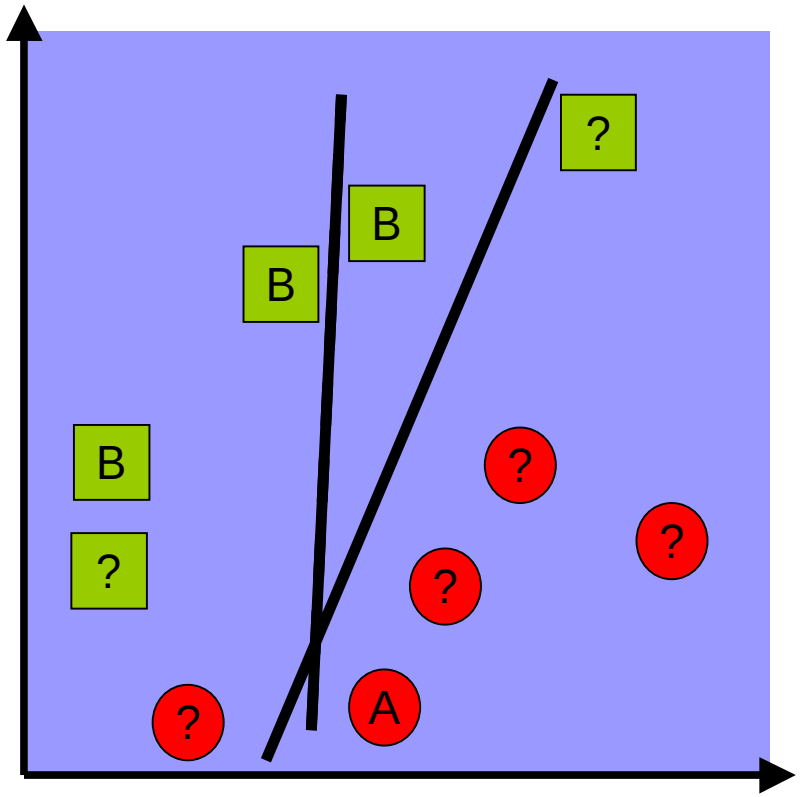
Active Learning Example

1. Take unlabeled pool of data
2. Label some data (seed)
3. Label each point of which the algo is ***uncertain***



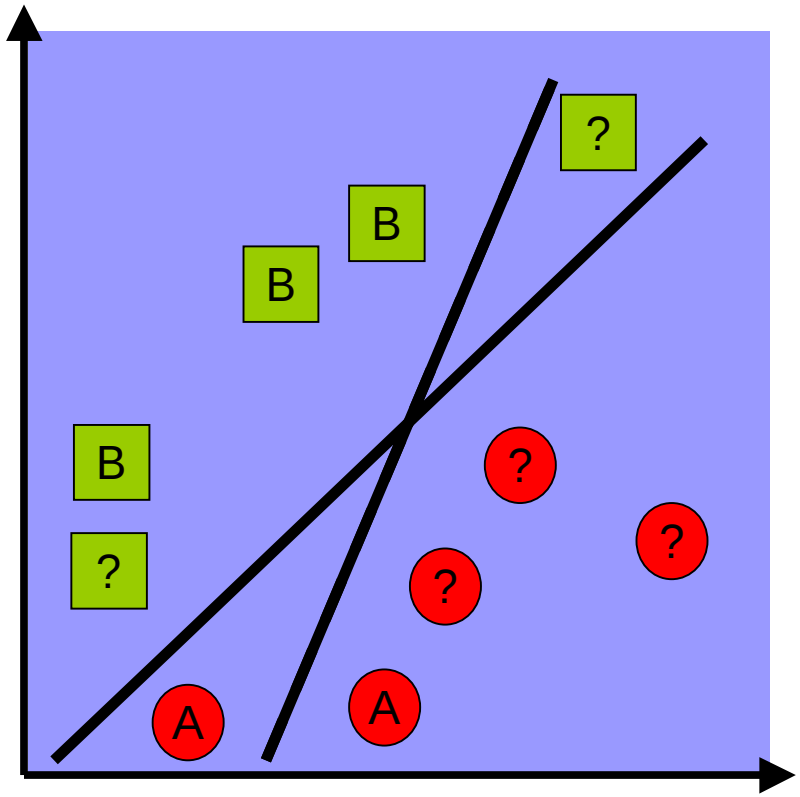
Active Learning Example

1. Take unlabeled pool of data
2. Label some data (seed)
3. Label each point of which the algo is ***uncertain***



Active Learning Example

1. Take unlabeled pool of data
2. Label some data (seed)
3. Label each point of which the algo is ***uncertain***



Outline

- Background
- Introduction to *Active Learning*
- Experiment Details
- Results
- Conclusion

Comparison of an AL with a traditional self-learning system

- Explore benefits active learning may bring to intrusion detection.
- Data used for experiment
  - Modified 10% KDD data
    - originally Lincoln Labs,
    - preprocessed by Columbia
    - Data chosen despite critique, as we wanted others to be able to replicate our work.

Algorithms used in experiment

- Mainly used two different “active learners.”
  - *Simple active learner*
    - *f* is a support vector machine (SVMLight)
    - *q* is the simple query function described earlier.
  - *Random learner* (used as a reference)
    - *f* is a support vector machine.
    - *q* is a function that chooses examples randomly from *P*.

Experiment Series

Exp	*Data set*	*Pool Size*	*Attacks*
1	Neptune	200,1000, 5000	50%
2	Neptune	200,1000, 5000	1%
3	Normal	200, 1000	50%

- **Neptune**= benign & DoS events from *neptune*
  - Manifested on network level (“easy” detection).
- **Normal** = benign & all attack events
  - “Complex” and difficult to classify correctly.

Metrics Used

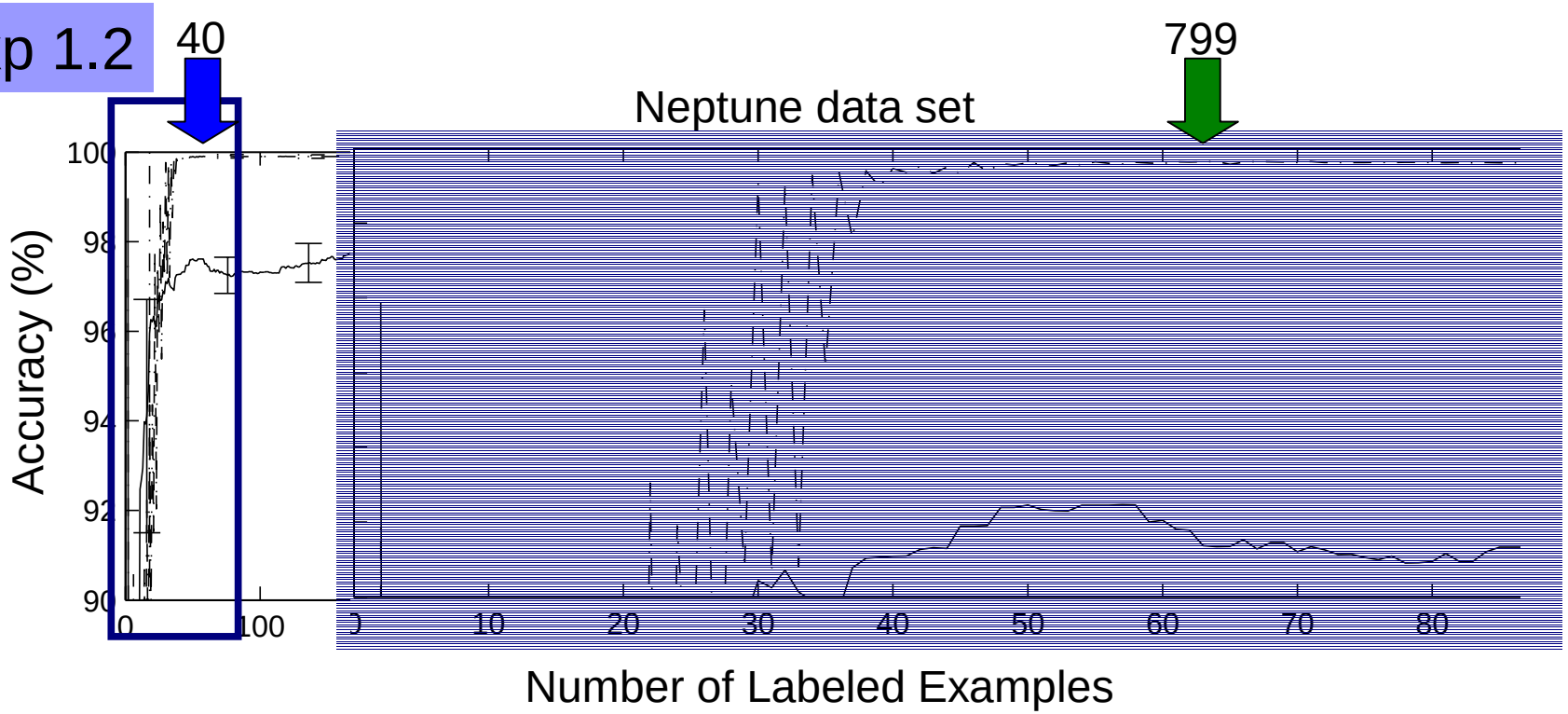
- **Accuracy:** $$\frac{TN + TP}{TN + FN + TP + FP}$$
- **Stable Point:** Current accuracy *remains* greater than a certain limit (here chosen to be 0.1%) of the final accuracy of the run.
- **Random Catchup:** Point where random and simple learner equivalent (95% sign)

More metrics described in paper.

Outline

- Background
- Introduction to *Active Learning*
- Experiment Details
- Results
- Conclusion

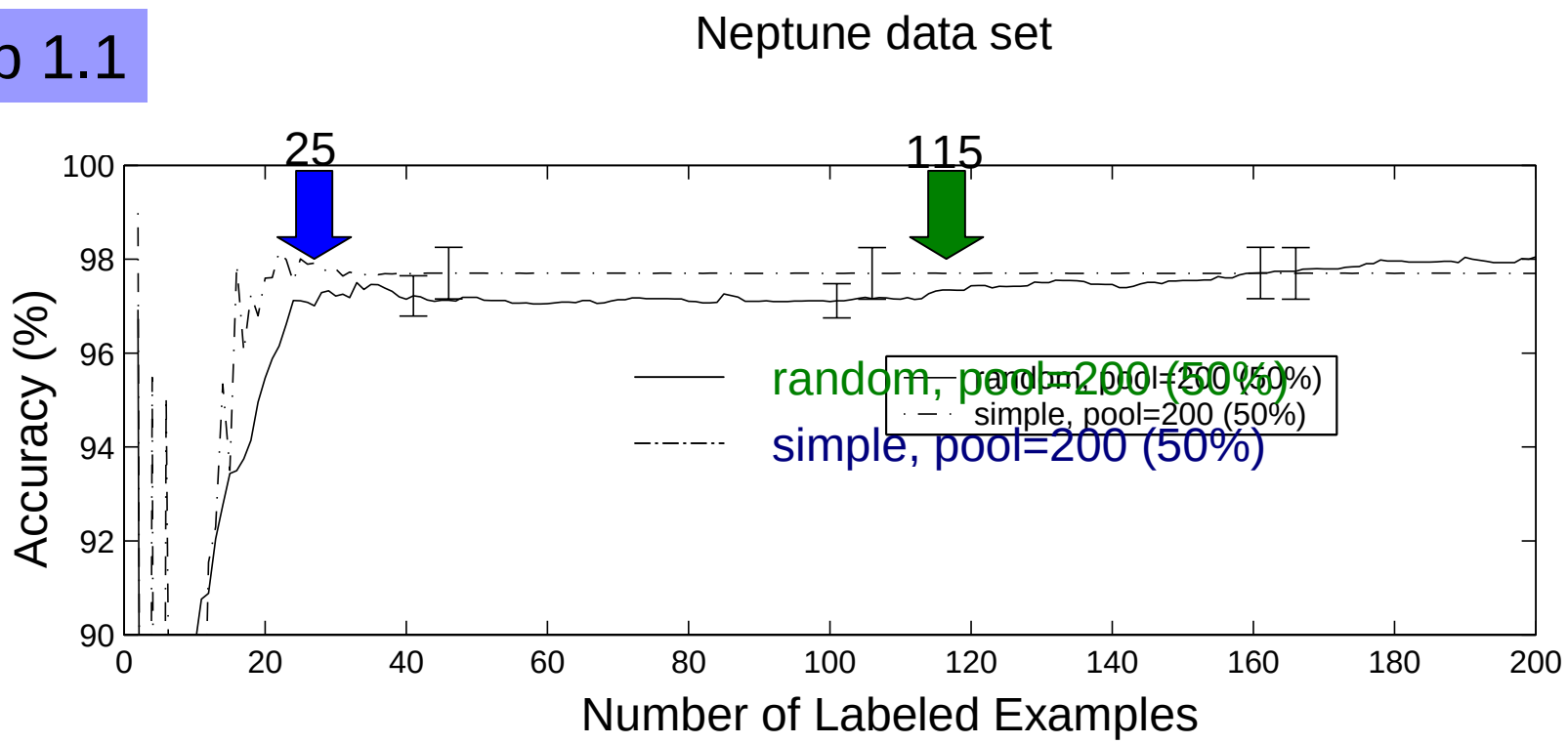
Exp 1.2



- Final accuracy: 99.90%
- Stable point of AL classifier: 40
- Random catchup of ref classifier: 799

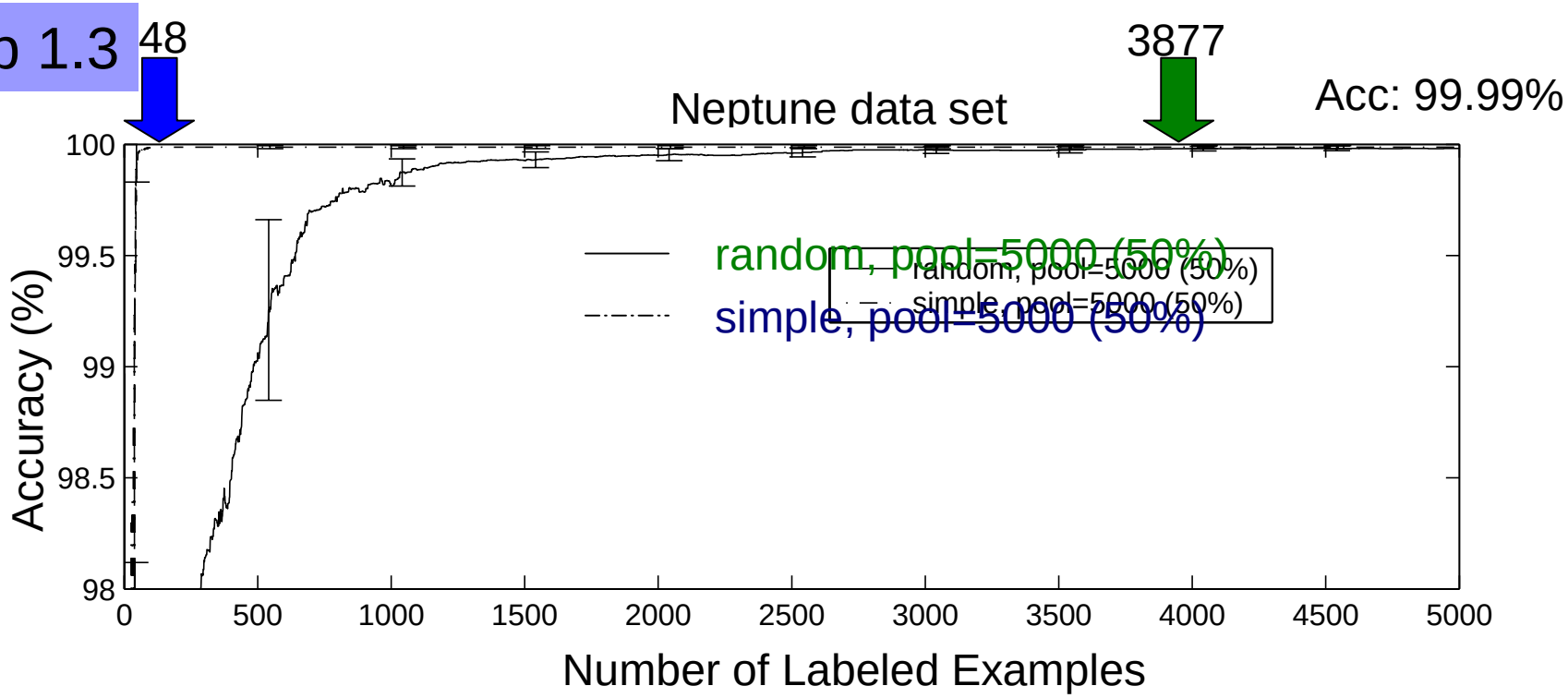
Reduction of labeled examples by 20 times.

Exp 1.1



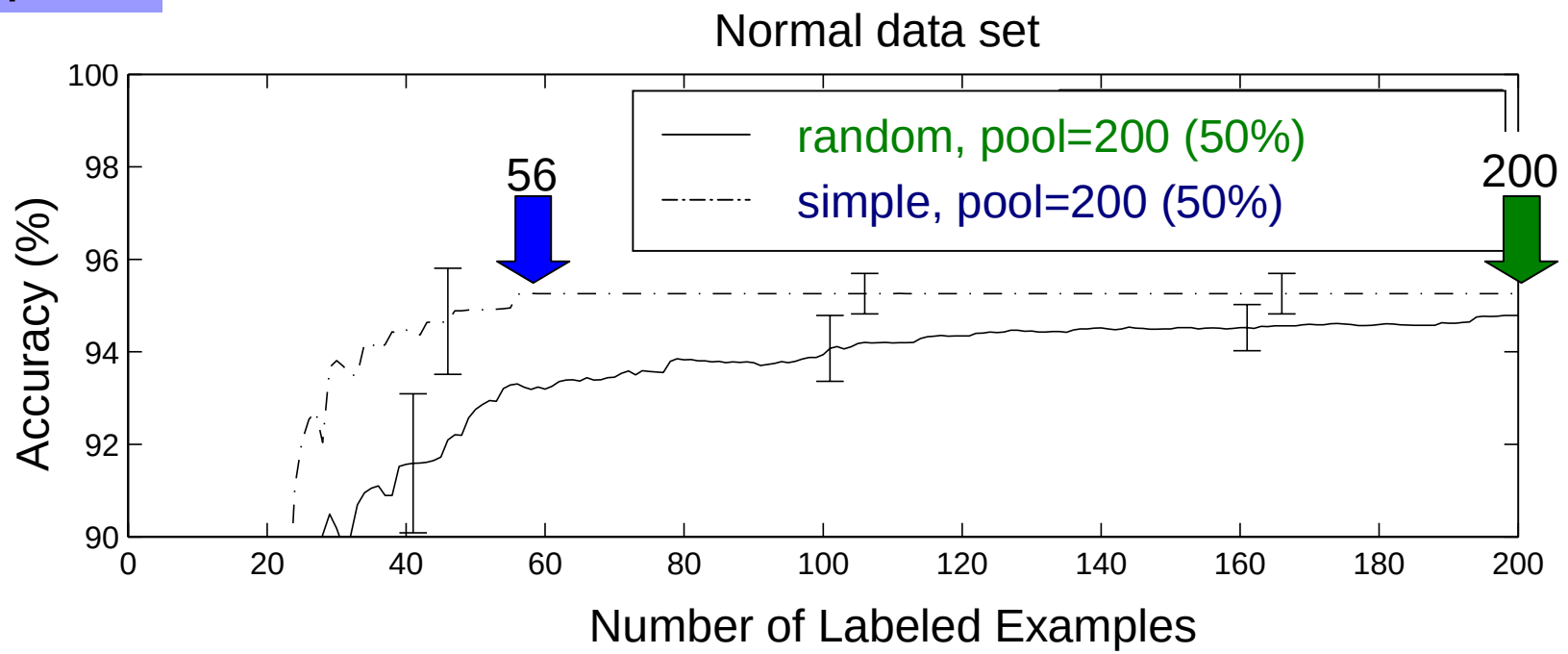
- Accuracy overall lower (97.70%) (200 example pool too small).
- Active learner 4.5 times more effective than ref learner.

Exp 1.3



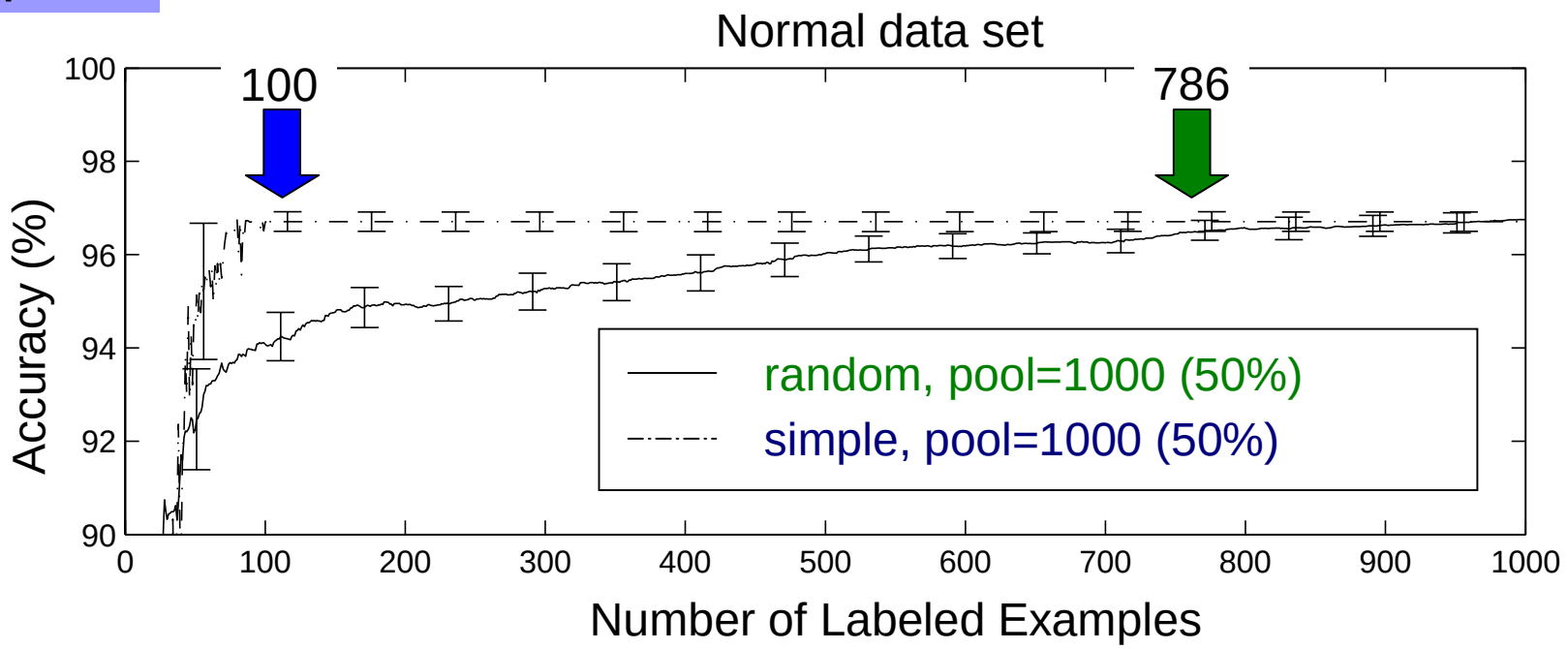
- Very good accuracy (only a few misclassifications).
- Ref learner needs 80 times as many labeled examples as compared to the active learner.
- Accuracy for the last 1000 examples remains the same (sufficient pool size?)

Exp 3.1



- Final accuracy: 95.26%
- Difficult set to classify correctly, and more support vectors are needed.
- Ref learner needs more than 3.5 times as many labeled examples as compared to the active learner.

Exp 3.2



- Final accuracy: 96.71%
- Ref learner needs 8 times as many labeled examples as compared to the active learner.

Outline

- Background
- Introduction to *Active Learning*
- Experiment Details
- Results
- Conclusions

Conclusion

- AL reduces the number of examples an expert needs to label.
  - Active learner still performs on a par with a traditional learner.
  - Success depends on pool size and “complexity.”
  - If label reduction can be directly be translated into saved time, we have in the best case:

1 hour work instead of 2 weeks

(which may in turn open up new apps for IDS).

Questions?