On the Effects of Learning Set Corruption in Anomaly-based Detection of Web Defacements

Eric Medvet and **Alberto Bartoli**

DEEI, University of Trieste, Italy

Anomaly detection in brief

- Common approach for intrusion detection
- How it works:
  1. Builds a profile of the resource to be monitored based on a **learning set**
  2. Signals **deviations** from the profile
     - **Assumption**: any deviation represents an evidence of an attack

Anomaly detection: requirements

- Learning set ≡ "**normal**" usage of the resource
  - It should be representive enough
  - It should be **attack-free**

Attack free learning set

- Usually the set is checked “manually” or taken for clean
- Problematic if:
  - Many resources to be monitored
  - Need to update profiles over the time

Coping with corrupted learning set

1. **Understanding** how it affects anomaly detection effectiveness
2. **Detecting**, i.e., being able to discriminate between corrupted and clean learning sets
3. **Mitigating**, i.e., preserving anomaly detection effectiveness in spite of corruption

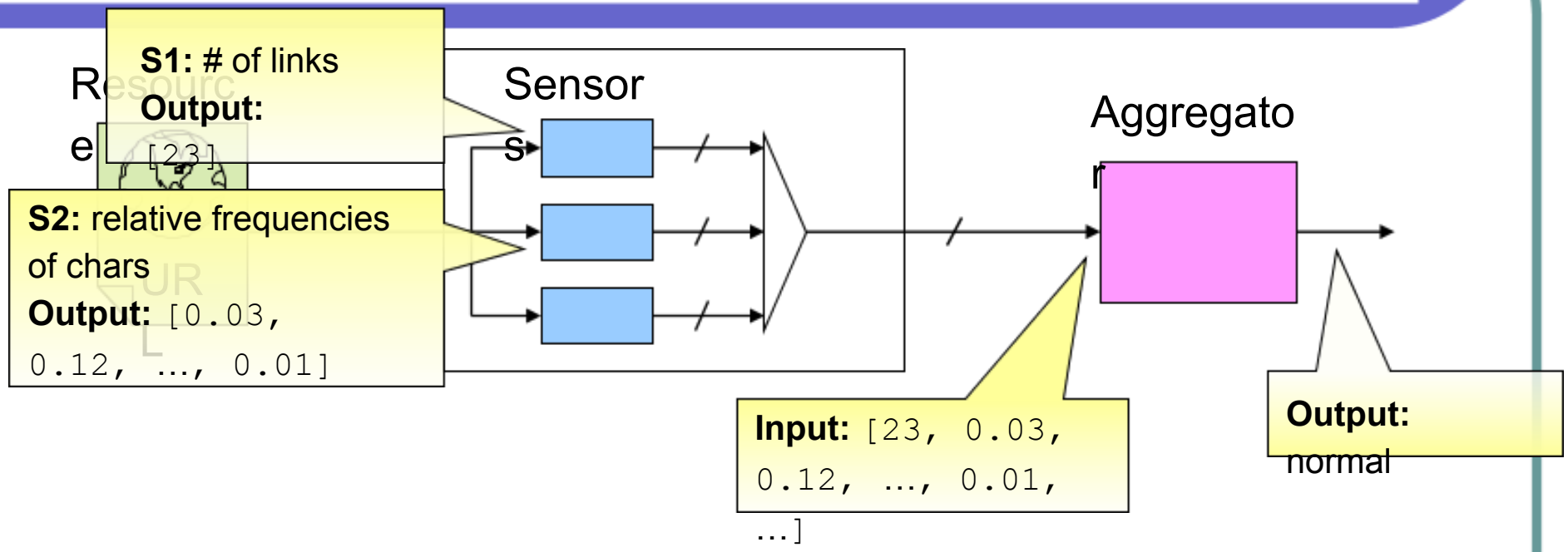
Our case study

- **Anomaly-based defacement detection** system that we developed earlier
  (IEEE Internet Computing Nov-Dec 2006)
- Monitors **many remote** web pages at regular intervals and raises an alert when a page deviates from its profile
- Key features:
  - **Dynamic** pages
  - **No prior knowledge** about the monitored pages

Attack-free?

- Exactly the “problematic” scenario:
  - Need to update profiles over the time
  - Many resources to be monitored
- Moreover:
  - Learning sets are small (~30 readings)
  - Even a single corrupting reading might be too much...

Checker: how it works

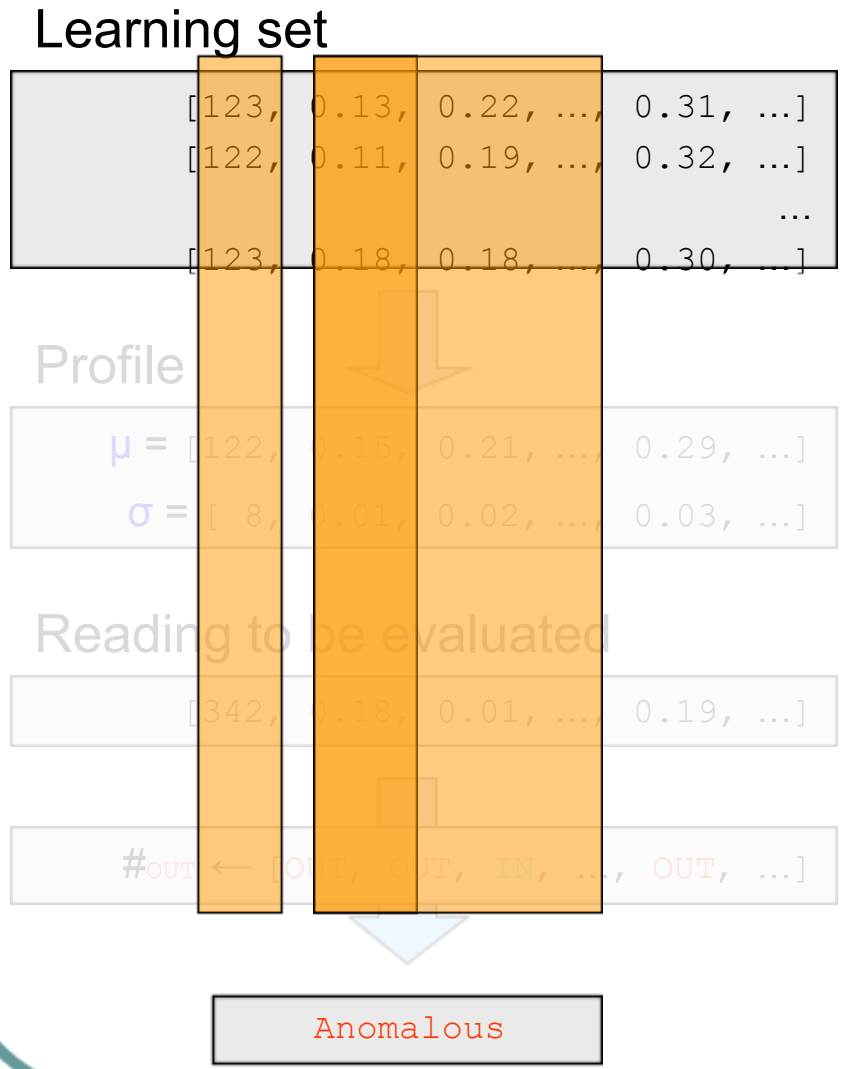


- Sensor
  - Functional block
  - Quantifies page features
- Aggregator
  - Builds the profile
  - Detects deviations from profile

Sensors

- In our prototype
  - 43 sensors
  - Grouped in 5 “categories”
  - Producing a numerical vector with 1466 elements
- “Many” page features from “many” points of view
  - Details in the paper

Aggregators in brief



- For the **simplest aggregator**, profile is composed by μ and σ
- Evaluations: check for $v_i \notin [\mu_i - 3\,\sigma_i, \mu_i + 3\,\sigma_i]$
- The other involves more complex operations, grouping vector elements in a different way

Aggregators

- We evaluated
  - Varying amount of corruption in the learning set
  - With 3 different aggregators
- Example (simplest aggregator):
  - Detects deviations from mu±3 sigma separately for each vector element
  - Sensitivity ≡ Number of “anomalous” elements

Checker effectiveness indexes

- False positive rate (**FPR**), false negative rate (**FNR**) and area under the ROC curve (**AROC** or AUC)
- Given a page, a set S_N of its genuine readings and a set S_P of positive readings (attacks)
  - Use part of S_N to build the profile
  - Use the remaining to test for false positives
  - Use S_P to test for false negatives

Experiments: data set

- Negative readings
  - **15 web pages** observed for…
  - …**1 year** and downloaded…
  - …each **6 hours**
  - totalling about 20000 readings
- Positive readings
  - **100 readings** extracted from a publicly available attacks (defacements) archive

Experiments: in a nutshell

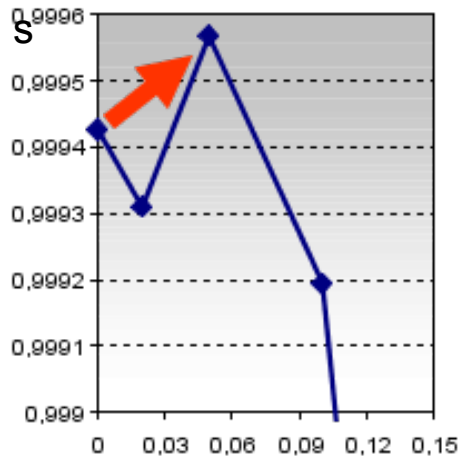
- We built a number of **corrupted** learning sets
- Varying values for the corruption rate **r**
  - 2%, 5%, …, 75%
- Experiments repeated for each of the 3 aggregators
- **Goal:** measure how r affects FPR, FNR and AROC for the given aggregator

Experiments: some details

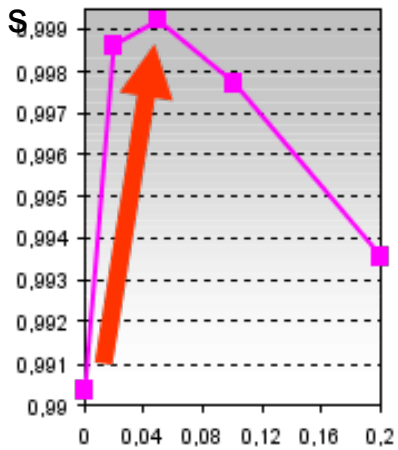
- We started with a set S_n of genuine readings
- We inserted in it one (repeated) simulated attack p extracted from S_p
- We evaluated FPR, FNR, A_{ROC} for the resulting corruption rate r
- Repeated by varying r *(2%, 5%,… 75%)*
  - Repeated by varying p, S_n and page, totalling 1500 tests for each r
- Repeated all the above for each of the 3 aggregators

Effects on corruption rate on AROC

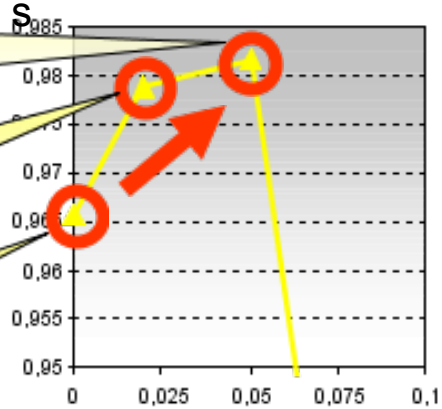
TooManyFiringGroups



TooManyFiringSensors



TooManyFiringInputs



Learning set with **3** **positive** reading

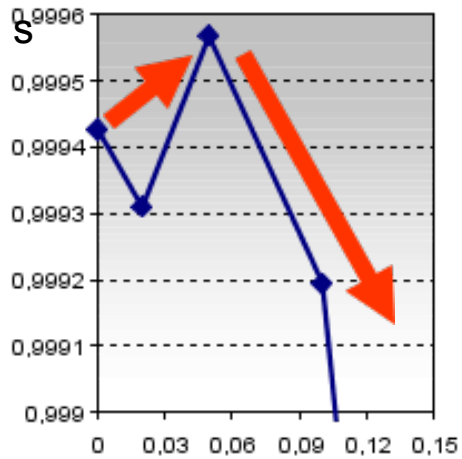
Learning set with **1** **positive** reading

Uncorrupted learning sequence

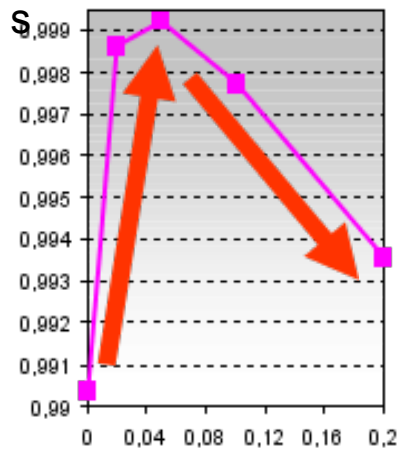
Effects on corruption rate on AROC

- AROC increases for small values of r…
- … and decreases for larger r
- This could be useful for *mitigation*…

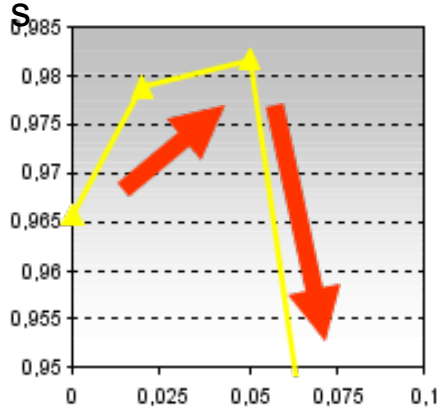
TooManyFiringGroup



TooManyFiringSensor

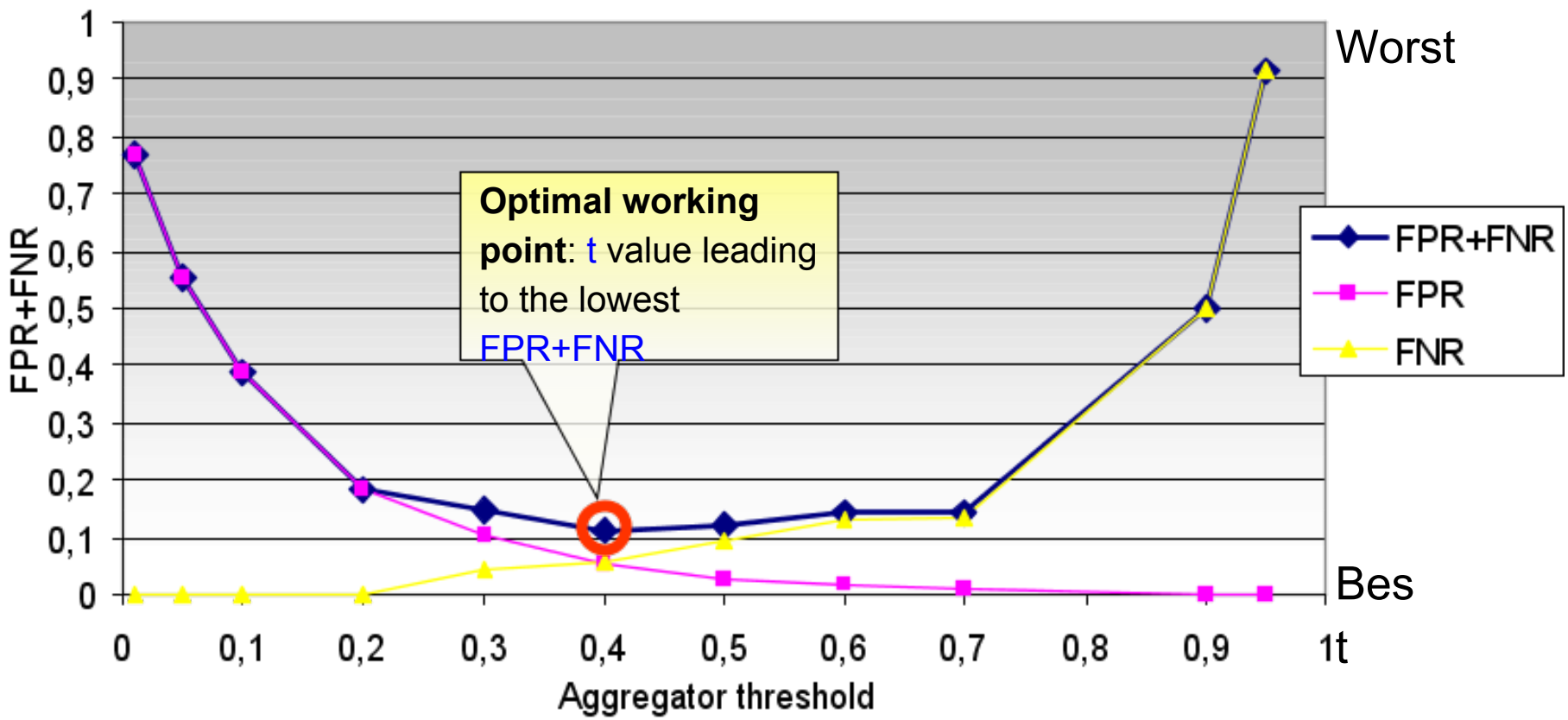


TooManyFiringInput



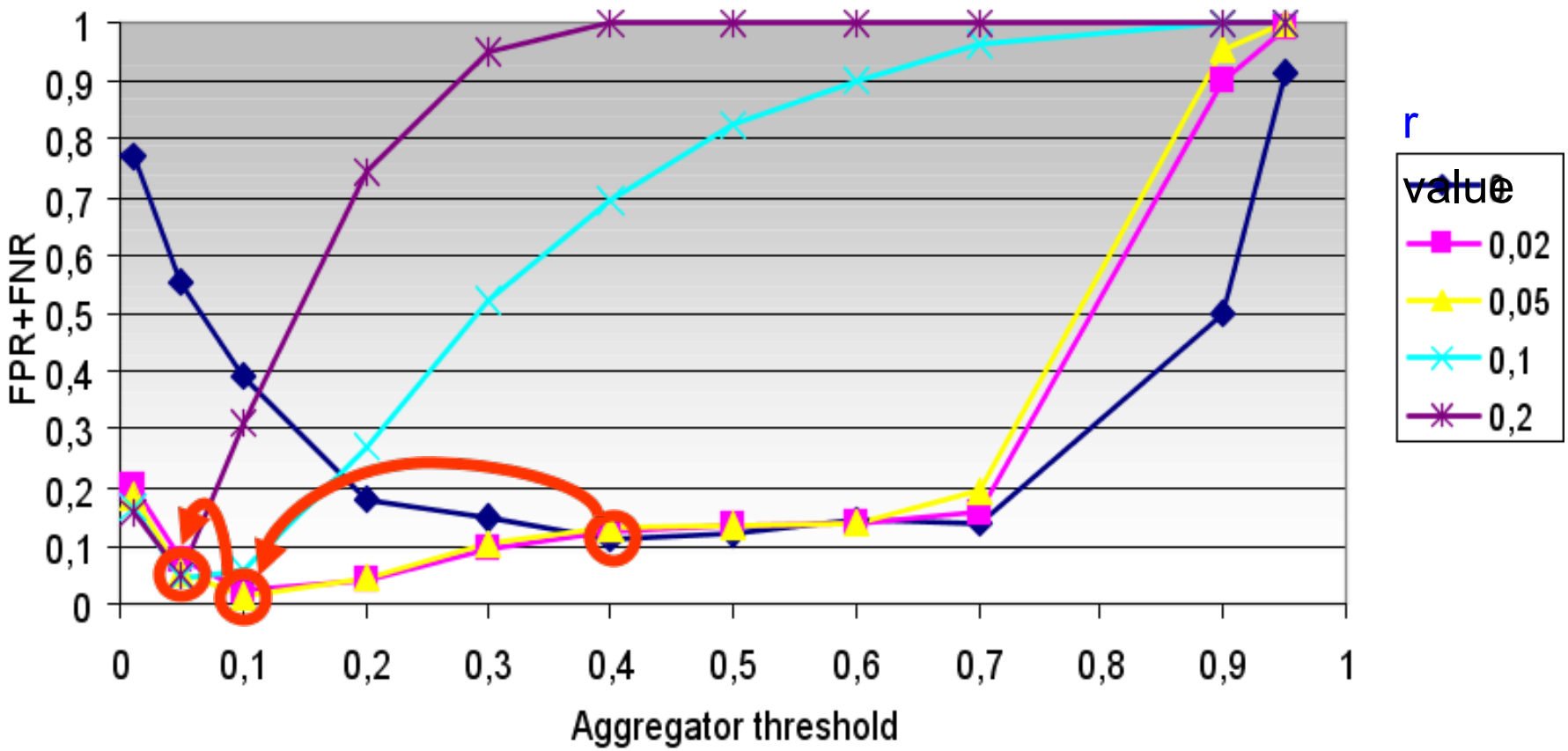
FPR and FNR with **uncorrupted** learning set

TooManyFiringSensors: FPR+FNR



FPR and FNR with **corrupted** learning set

TooManyFiringSensors: FPR+FNR



Understanding: conclusions

- As expected:
  - If r ↑ then FPR ↓ and FNR ↑
  - Optimal working points change
- **Interestingly:**
  - The more effective the aggregator, the greater the worsening
  - With a fixed t, corruption heavily affect FNR; changing t might avoid or reduce worsening
  - With low r values, effectiveness does not worse significantly in terms of A_{ROC}, instead it can improve

Detecting

- **Goal**: determining if a learning sequence S is clean or corrupted
- Difficult if:
  - Small learning set than cannot be further reduced
  - No hypothesis about the expected effectiveness

Detecting: key idea

1. We compute FPR and FNR using S
2. We artificially corrupt S with 1 and 3 positive readings, obtaining S_1 and S_3
3. We compute FPR and FNR using S_1 and S_3
4. If effectiveness with S is much better than with S_1 and S_3, then S was clean, otherwise, it was corrupted

- We need **not to divide** S in subsets
  - Works with small learning sets

Detecting: results

- Our procedure works quite well
  - Always detects corrupted sets, with r up to 0.5
- With the more effective aggregator
  - Our procedure works perfectly (no false positive, no false negatives)
- With the two other aggregators
  - It detects all corrupted sets, but shows a too high number of false positives

Detecting: motivation

- How r affects FNR and FPR on fixed t point?

	roups		TooManyFiring		nyFiringInputs	
	FPR		FPR		FPR	FNR
r	$t = t_{\text{opt}} = 0.9$		$t = t_{\text{opt}} = 0.4$		$t = t_{\text{opt}} = 0.05$	
0.00	0.0	1.9	5.4	5.7	11.4	5.2
0.02	0.0	73.4	0.1	12.4	6.5	5.6
0.05	0.0	77.6	0.1	12.9	5.5	6.3
0.10	[illegible]	83.9	0.0	69.5	2.5	75.8
0.20	[illegible]	[illegible]	0.1	99.9	2.5	99.4
0.35	[illegible]	[illegible]	0.1	99	2.6	99.6
0.50	[illegible]	[illegible]	0.1	[illegible]	.8	99.6
0.75	0.0	87.7	0.3	[illegible]	11.3	99.4

From clean to modestly corrupted sequence

Significant FPR and/or FNR variations

From modestly corrupted to a bit more corrupted

Much less amount of variation

- Thanks!