TTCN-3 Quality Engineering: Using Learning Techniques to Evaluate Metric Sets

Edith Werner, Jens Grabowski, Helmut Neukirchen, Nils Röttger, Stephan Waack, and Benjamin Zeiss

Institute for Computer Science
University of Göttingen

Outline

- Motivation
- Evaluating Metric Sets
- TTCN-3 Experiments
- Summary and Outlook

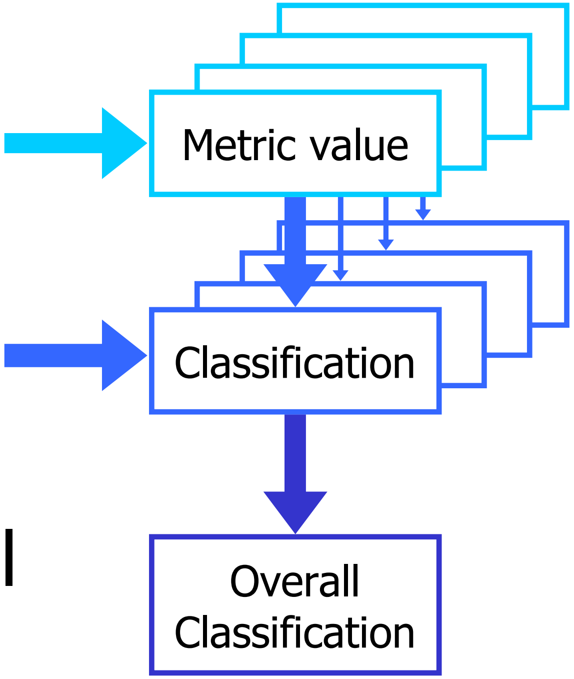
Motivation

"To measure is to know."

Lord William Kelvin, 1824-1907

Metrics

- A metric captures a quality aspect of the software
  - Computation rule
    - Counting
    - Basic arithmetics
  - Threshold
- Metrics are combined into quality models to capture all relevant quality aspects



Metric Sets

- Tradeoff:
  - Less metrics, less information
  - More metrics, more costs
- Ideally:
  - Measure as much as necessary but as little as possible.

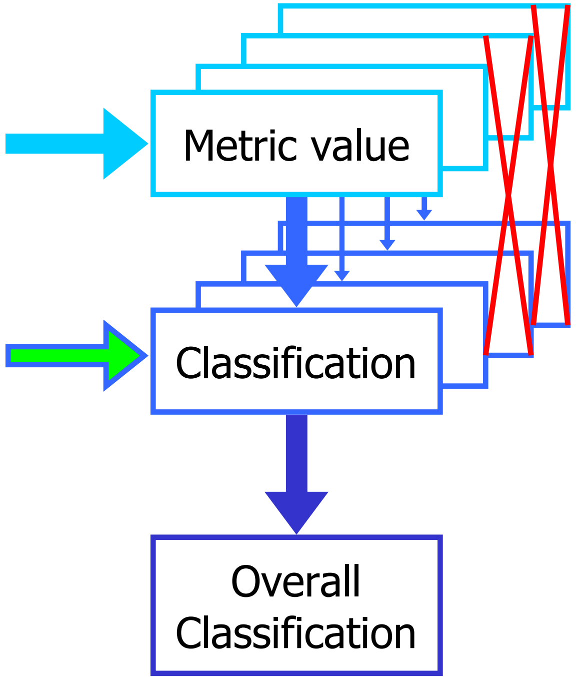
- How to determine an optimized metric set?

Outline

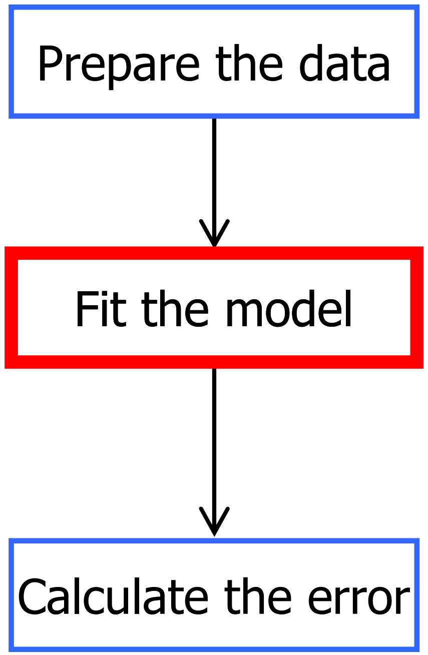
- Motivation
- Evaluating Metric Sets
- TTCN-3 Experiments
- Summary and Outlook

Basic Idea

- Given: a set of metrics with thresholds
- Find a subset that yields the same overall classification
- Adapt the metrics thresholds to achieve this



Learning Procedure


Prepare the data

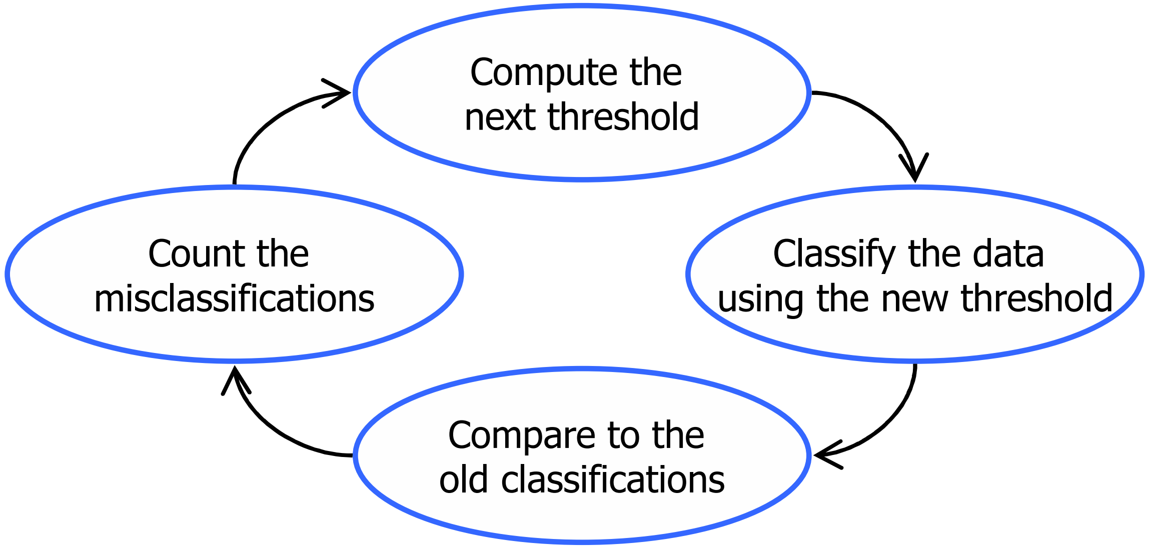
Fit the model

Calculate the error

- Select software entities and classify them using the original metric set
- Randomly divide the data into three sets
- Training set
  - Generate adapted threshold for each possible metric subset
- Validation set
  - Compute the error of the adapted metric subsets
- Test set
  - Compute the error of the best subset

Fit the Model - Example

- Approximate metric set by one metric


Compute the next threshold
Classify the data using the new threshold
Compare to the old classifications
Count the misclassifications

  - For each metric, choose the threshold yielding the least misclassifications
  - Repeat for every metric

Outline

- Motivation
- Evaluating Metric Sets
- TTCN-3 Experiments
- Summary and Outlook

Data

- Testing and Test Control Notation Version 3
  - Session Initiation Protocol (SIP) test suite
  - Internet Protocol Version 6 (IPv6) test suite
- TTCN-3 Metrics
  - Number of statements
  - Cyclomatic complexity
  - Maximum Nesting Level
  - Maximum Call Depth

Scenarios

- Scenario 1: Strict Classification
  - All four metrics must classify the software as "good"
- Scenario 2: Relaxed Classification
  - At least three of the metrics must classify the software as "good"
- Metric subsets
  - One of four
  - Two of four

Experimental Results

- The approach works
- Case Study
  - A set of four metrics can be approximated by a subset of two metrics with a test error of 1.94 %
  - The best approximation uses the threshold values that were used to generate the original classification
- Negative data is needed to prevent overfitting

Outline

- Motivation
- Evaluating Metric Sets
- TTCN-3 Experiments
- Summary and Outlook

Summary and Outlook

- Summary
  - Learning techniques can be used to optimize metric sets
  - The approach is applicable to TTCN-3
- Outlook
  - Apply the technique to Java and UML
  - Optimization of the algorithm for usage with larger metric sets

- Thank you for your attention!