Active Learning for Automatic Classification of Software Behavior

Jim Rehg
James F. Bowring
Mary Jean Harrold

Funded by: Office of Naval Research through a NDSEG Fellowship,
NSF awards CCR-9988294, CCR-0096321, CCR-0205422,
SBE-0123532, EIA-0196145,
State of Georgia Yamacraw Mission
My Collaborators

Mary Jean Harrold
NSF ADVANCE Professor
College of Computing

Jim Bowring
Ph.D. Candidate
College of Computing

http://www.cc.gatech.edu/aristotle/
Use Machine Learning Techniques to Characterize Program Behavior

• A program is ultimately the collection of its executions.
  ○ This collection is diverse, unpredictable, unbounded
  ○ Depends upon input data and changing needs of users.

• Behavior characterization
  ○ Formal methods (during requirements analysis)
  ○ Reliability assessment (during testing)

• Statistical characterization based on execution data
Motivation

Automatic classification could help in

• Construction of test oracles
• Detection of operational profiles

Thus extending the automation of

• Testing
• Program self-awareness

Basic approach

1. Extract statistical models from executions
2. Learn classifiers from these models
Our Contribution

Feature selection
Show that event transitions can be good predictors of behavior.

Model selection
Show that Markov models of event transitions can be clustered into effective classifiers.

Machine learning
Show how active-learning techniques can aid in the construction of behavior classifiers.
Presentation Overview

- Markov models from executions
- Our technique for automating classification
  - Building a classifier
  - Classifying an execution

- Test suite augmentation
- Empirical studies

- Related work
- Summary and research directions
Markov Models from Executions
Markov Models from Executions

Diagram:

Entry → p1 (T) → p2 (T, F) → s1 (T, F) → s3 → Exit

- Branch B1: Profile = 9
- Branch B2: Profile = 1
- Branch B3: Profile = 3
- Branch B4: Profile = 8
Markov Models from Executions
Markov Models from Executions

![Diagram of Markov Model with states and transitions]

Markov Model of Branch Profiles

<table>
<thead>
<tr>
<th></th>
<th>p1</th>
<th>p2</th>
<th>s1</th>
<th>s2</th>
<th>Exit</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>p2</td>
<td>9/10</td>
<td>0</td>
<td>3/9</td>
<td>6/9</td>
<td>0</td>
</tr>
<tr>
<td>s1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>s2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>s3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Markov Models from Executions

Markov Model of Branch Profiles

<table>
<thead>
<tr>
<th></th>
<th>p1</th>
<th>p2</th>
<th>s1</th>
<th>s2</th>
<th>Exit</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1/10</td>
</tr>
<tr>
<td>p2</td>
<td>0</td>
<td>0</td>
<td>3/9</td>
<td>6/9</td>
<td>0</td>
</tr>
<tr>
<td>s1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>s3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Branch B1: Profile = 9
Branch B2: Profile = 1
Branch B3: Profile = 3
Branch B4: Profile = 6
Building a Classifier

Stage 1. Prepare Training Instances

Program P

Training Instances = Event Transition Profiles w/ Behavior Labels

Stage 2. Train Classifier

Test Suite w/ Behavior Oracle

Classifier C
Building a Classifier

Stage 1. Prepare Training Instances

Stage 2. Train Classifier

Training Instances = Event Transition Profiles w/ Behavior Labels

Program P

Test Suite w/ Behavior Oracle

Classifier C
Building a Classifier

Stage 1. Prepare Training Instances

Stage 2. Train Classifier

Training Instances = Event Transition Profiles w/ Behavior Labels

Program P

Test Suite w/ Behavior Oracle

Classifier C
Building a Classifier

Stage 1. Prepare Training Instances

Test Suite w/ Behavior Oracle

Training Instances = Event Transition Profiles w/ Behavior Labels

Stage 2. Train Classifier

Program P

Classifier C
Stage Train Classifier

Group by Behavior Labels

Behavior Groups: b1, ..., bn

Train one Classifier per Group

Classifiers: Cb1, ..., Cbn

Assemble Classifier C for P

Classifier C
Stage Train Classifier

- Group by Behavior Labels
- Behavior Groups: $b_1, \ldots, b_n$
- Train one Classifier per Group
- Classifiers: $C_{b1}, \ldots, C_{bn}$
- Assemble Classifier $C$ for $P$

Classifier $C$
Stage Train Classifier

Group by Behavior Labels

Behavior Groups b1,...,bn

Train one Classifier per Group

Classifiers Cb1,...,Cbn

Assemble Classifier C for P

Classifier C
Stage Train Classifier

Group by Behavior Labels

Behavior Groups b1, ..., bn

Train one Classifier per Group

Classifiers Cb1, ..., Cbn

Assemble Classifier C for P

Classifier C
Training a Classifier

Clustering Example:

- behavior label “pass”
- initially contains 16 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- initially contains 16 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- … now contains 15 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- … now contains 14 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- … now contains 13 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- … now contains 12 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- … now contains 11 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- … now contains 10 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- … now contains 9 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- … now contains 8 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- … now contains 7 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- … now contains 6 models
Training a Classifier

Clustering Example:

- behavior label “pass”
- … now contains 5 models
Training a Classifier

Clustering Example:

• behavior label “pass”
• ... now contains 4 models
• stopping criterion
Classifying an Execution

Markov Model of Branch Profiles

<table>
<thead>
<tr>
<th></th>
<th>p1</th>
<th>p2</th>
<th>s1</th>
<th>s2</th>
<th>Exit</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>0</td>
<td>9/10</td>
<td>0</td>
<td>0</td>
<td>1/10</td>
</tr>
<tr>
<td>p2</td>
<td>0</td>
<td>0</td>
<td>3/9</td>
<td>6/9</td>
<td>0</td>
</tr>
</tbody>
</table>
Classifying an Execution

New Execution E of Program

Markov Model of Branch Profiles

<table>
<thead>
<tr>
<th></th>
<th>p1</th>
<th>p2</th>
<th>s1</th>
<th>s2</th>
<th>Exit</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>0</td>
<td>9/10</td>
<td>0</td>
<td>0</td>
<td>1/10</td>
</tr>
<tr>
<td>p2</td>
<td>0</td>
<td>0</td>
<td>3/9</td>
<td>6/9</td>
<td>0</td>
</tr>
</tbody>
</table>
Classifying an Execution

Profile: \{ B1 ^, B3 ^, B2 \}

\[
P(M \text{ produced } E) = (\frac{9}{10})^2 \times (\frac{3}{9})^2 \times (\frac{1}{10})
\]

\[
= 0.008982
\]
Presentation Overview

- Markov models from executions
- Our technique for automating classification
  - Building a classifier
  - Classifying an execution

- Test suite augmentation
- Empirical studies

- Related work
- Summary and research directions
Test Suite Augmentation

Stage 1. Prepare Training Instances

Training Instances = Event Transition Profiles w/ Behavior Labels

Stage 2. Train Classifier

Program P

Test Suite w/ Behavior Oracle

Classifier C
Test Suite Augmentation

Program P

Stage 1. Prepare Training Instances

Training Instances = Event Transition Profiles w/ Behavior Labels

Test Suite w/ Behavior Oracle

Stage 2. Train Classifier

Classifier C

Test data generator

Execute

Hand Classify Test case
Test Suite Augmentation

Batch learning to refine the classifier
Test Suite Augmentation

Program P

Stage 1. Prepare Training Instances

Training Instances = Event Transition Profiles w/ Behavior Labels

Stage 2. Train Classifier

Classifier C

Execute and Classify

Test data generator

Test Suite w/ Behavior Oracle

Hand Classify Test case

Known

Unknown Behavior
Test Suite Augmentation

Program P

Stage 1. Prepare Training Instances

Training Instances = Event Transition Profiles w/ Behavior Labels

Stage 2. Train Classifier

Test data generator

Execute and Classify

Classifier C

Hand Classify Test case

Test Suite w/ Behavior Oracle

Active learning
to refine the classifier

Known

Unknown Behavior
Empirical Studies

Studies: Measure quality of classifier refinement with
1. Batch learning  2. Active learning

Measures:

Acceptance Rate = (# Classified) / (# Attempted)
Classification Rate = (# Correctly Classified) / (# Classified)

Subject: Space (7 KLOC C) + 13585 tests

Method: For each of 15 versions of Space, repeat 10 times

1) Build initial classifier from 100 random test cases
2) Cross validate the classifier on remaining test cases
3) Repeat 15 times:
   a) Refine classifier with additional using batch or active
   b) Cross validate classifier on remaining test cases
Transitions

- We have explored several different transitions for the execution data
  - Branch (as in the previous example)
  - Branch to branch (a 2nd order statistic)
  - Methods (transitions between method calls)

- Results
  - Aggregate results for branch transitions from SPACE
  - Preliminary results for all 3 on flex, grep, print_tokens, schedule, …
Study Batch Learning

Acceptance Rate

Detection Rate vs Training Set Size graph.
## Study Batch Learning

### Classification Rate

<table>
<thead>
<tr>
<th>Training Set Size</th>
<th>Number of Classifiers</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>150</td>
<td>0.976</td>
<td>0.0449</td>
</tr>
<tr>
<td>150</td>
<td>150</td>
<td>0.974</td>
<td>0.0435</td>
</tr>
<tr>
<td>200</td>
<td>150</td>
<td>0.975</td>
<td>0.0416</td>
</tr>
<tr>
<td>250</td>
<td>150</td>
<td>0.976</td>
<td>0.0399</td>
</tr>
<tr>
<td>300</td>
<td>150</td>
<td>0.977</td>
<td>0.0353</td>
</tr>
<tr>
<td>350</td>
<td>150</td>
<td>0.976</td>
<td>0.0321</td>
</tr>
</tbody>
</table>
Study  Active Learning

Acceptance Rate

![Graph showing the relationship between Detection Rate and Training Set Size. The graph indicates a positive correlation, with the Detection Rate increasing as the Training Set Size increases.]
### Study: Active Learning

**Classification Rate**

<table>
<thead>
<tr>
<th>Training Set Size</th>
<th>Number of Classifiers</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>150</td>
<td>0.979</td>
<td>0.0429</td>
</tr>
<tr>
<td>150</td>
<td>150</td>
<td>0.979</td>
<td>0.0435</td>
</tr>
<tr>
<td>200</td>
<td>150</td>
<td>0.977</td>
<td>0.0467</td>
</tr>
<tr>
<td>250</td>
<td>150</td>
<td>0.976</td>
<td>0.0463</td>
</tr>
<tr>
<td>300</td>
<td>150</td>
<td>0.978</td>
<td>0.0396</td>
</tr>
<tr>
<td>350</td>
<td>150</td>
<td>0.979</td>
<td>0.0358</td>
</tr>
</tbody>
</table>
Batch vs. Active

Acceptance Rate

Detection Rate vs. Training Set Size

Active

Batch
Preliminary Results for flex

Acceptance

Branch

Branch to Branch

Classification

Method
Presentation Overview

✓ Markov models from executions
✓ Our technique for automating classification
  ✓ Building a classifier
  ✓ Classifying an execution
✓ Test suite augmentation
✓ Empirical studies

Related work
Summary and research directions
Related Work

Automated Clustering
Dickinson, Leon, and Podgurski (ICSE ’01)
Podgurski and colleagues (ICSE ’03)

Statistical Learning
Brun and Ernst (ICSE ’04)
Gross and colleagues (RAMSS ’03)
Harder, Mellen, Ernst (ICSE ’03)
Lin and Ernst (ISSTA ’04)
Munson and Elbaum (HICSS ’99)

Markov Models
Cook and Wolf (ICSE ’95)
Whittaker and Poore (TOSEM Jan ’96)
Summary and Research Directions

Summary

- Technique to automatically construct classifiers from Markov models of event transitions
- Empirical results that indicate the viability of our approach in concert with active learning

Future work

- Subjects: more subjects, multiple faults
- Features: order transitions, data-flow transitions
- Behaviors: model specific sub-behaviors
- Learning: Expectation-maximization, Hidden Markov models