Today's software systems exceed millions of lines of code, with complex components and teams of developers working on the same application.

Software developers can be more effective in production and maintenance when they use tools, including those designed for:

- Searching Code
- Comprehending Code
- Identifying Bugs

**Overview of Approach**

**Leading Comment & Method Signature**

```c
/** Cancels the current HTTP request. */

void jsxFunction_abort(String fullName)
```

**Action Pair**

```c
/** supposed to go and locate offset of 
* the closing sequence within Str, or 
* npos if it was not found */

size_t FindRegexVarEnd(StringRef Str)
```

**Terminology**

**Descriptive Leading Comment**: A comment block placed before a method signature that provides the reader with an overall summary of a method's actions.

**Main Action**: A term that describes the collective intent of the method body.

**Phases of Automation**

**Methods with Leading Comments**

- Identify methods with descriptive leading comments
- Extract main action from descriptive comment
- Extract main action from associated signature
- Analyze & rank comment-code word pairs

**Method Signatures**

- Automatically generated semantically-similar word pairs

**Challenge II-a**: Extract main actions from leading comments

**Challenge II-b**: Extract main actions from method signatures (method names)

**Challenge III**: Generate semantically-similar pairs from data

**Evaluation: Methodology**

Overall, approximately 1.7 million methods with leading comments were analyzed from 14,904 open source programs written in Java and C++.

Human annotators with extensive knowledge of Java and C++ and no prior knowledge of the research were consulted to evaluate the results by analyzing a sampling of the raw input. The human-annotated results were then compared against the automated process for each phase to determine accuracy.

Each input was seen by 2 annotators, and a 3rd was brought in to settle disputes when they disagreed.

**Evaluation: Results**

The automatic miner was evaluated via 5 research questions:

**Q1**: How well does our automatic system determine descriptive comments?

**Result**: Automated system and humans agreed on 87.33% of comments.

**Q2**: How well does our automatic system identify the action word from a descriptive comment?

**Result**: Automated system agreed with at least 1 annotator on 91.96% of comments. When 2 or more annotators agreed on the action, our automated system agreed with them on 92.00% of comments.

**Q3**: How accurate is the automatic extraction of the main action from a method signature?

**Result**: Automated system agreed with at least 1 annotator on 93.33% of signatures.

**Q4**: How well do we recall word-pairs from a human-annotated gold set?

**Result**: Automated system found 22 of 24 (91.67%) of pairs in a manually obtained gold-set.

**Q5**: How well do our word-pairs reflect reasonable semantically-similar words in computer science?

**Frequency**

<table>
<thead>
<tr>
<th>Frequency</th>
<th>% Rated Sem. Similar</th>
<th>% Rated Sem. Similar</th>
</tr>
</thead>
<tbody>
<tr>
<td>94+</td>
<td>87.80%</td>
<td>94.74%</td>
</tr>
<tr>
<td>67-93</td>
<td>66.97%</td>
<td>76.92%</td>
</tr>
<tr>
<td>50-66</td>
<td>53.85%</td>
<td>58.33%</td>
</tr>
<tr>
<td>40-49</td>
<td>68.75%</td>
<td>70.00%</td>
</tr>
</tbody>
</table>

**Table**: Sample Pairs Not Found in English Lexical Database (WordNet)

- init – initialize
- reset – clear
- fire – notify
- close – flush
- find – search
- update – refresh
- read – decode
- free – close
- parse – read
- purge – remove
- update – reload

**No Project Threshold**: Appear in >10 Projects

**Result**: Our automatic miner has high accuracy in all phases, including accuracy above 87% in identifying descriptive comments and extracting main actions from both comments and method signatures.

The automated system found expected pairs (as established by a gold set), and exhibited increasing reliability in determining semantic similarity for pairs with higher frequency. Further, it is possible to mine software-based, semantically-similar words from code and its documentation.