Recommending API Methods Based on Identifier Contexts

Lars Heinemann, Benjamin Hummel
Technische Universität München

SUITE 2011, Waikiki, Honolulu, Hawaii

Reuse Recommendation Systems

- Infer from current programming context what might be relevant artifacts for reuse
  - e.g. useful API methods or code examples
- Relieves developer from formulating query as needed for classic code search engines

Existing Approaches

- Use of syntactic structure of the program, e.g.:
  - method/type usage
  - inheritance relations
  - Problematic for code sections with general purpose types and without API calls:

```
if (angle != getAngle()) {
    float angleDelta = angle - getAngle();
    super.setAngle(angle);
```

Proposed Solution

- Focus on identifiers
- Embody valuable knowledge about developers‘ intent
- Here: trigonometry

```
if (angle != getAngle()) {
    float angleDelta = angle - getAngle();
    super.setAngle(angle);
```

➪ Suggestion of trigonometric API functions helpful

Proposed Approach

2 phases

- **Association Mining**:

  Learning term-method associations from software systems using the API

- **Recommendation Phase**:

  API method recommendations during development of new code

Association Mining

```
try {
    readFile();
}
catch (IOException e) {
    String message = e.getMessage();
    JOptionPane.showMessageDialog(null,
        message);
}
```

”lookback” = 2 lines

Identifier extraction, splitting, stemming

{io, except, string, messag, get}→
JOptionPane#showMessageDialog

Recommendation Query

```
try {
   readFile();
}
catch (IOException e) {
   String message = e.getMessage();
    |
```

”lookback” = 2 lines

Identifier extraction, splitting, stemming

{io, except, string, messag, get}

Index Query

- Find similar contexts (i.e. term sets) in index and recommend associated methods
- Use of vector space model
- Distance measure, e.g. hamming distance

Evaluation

- Try to predict method calls in existing open source code from preceding identifiers
- Mining associations from
  - other projects (cross-project recommendation)
  - parts of the same project (intra-project recommendation)

Findings

- On average, correct method recommended in every third case (5 recommendations)
- Recommendation rate decreases with increased lookback
  - Noise from unrelated identifiers
- Intra-project better than cross-project
  - Naming conventions

Future Work

- Inclusion of keywords
- Filtering of stopwords
- Filtering of well-known methods
- Quantitative comparison to existing approaches
- Other APIs (e.g. Eclipse)
- User study

Thank you.