When and How Using Structural Information to Improve IR-based Traceability Recovery

Annibale Panichella¹, Collin McMillan², Evan Moritz³, Davide Palmieri⁴, Rocco Oliveto⁴, Denys Poshyvaniyk³, Andrea De Lucia¹

¹ Software Engineering Lab, University of Salerno, Italy  
² University of Notre Dame, Notre Dame, USA  
³ The College of William and Mary, Williamsburg, USA  
⁴ University of Molise, Pesche (IS), Italy
Traceability Recovery is the ability to describe and follow the artifacts life-cycle.
Traceability Recovery is the ability to describe and follow the artifacts lifecycle.

- Impact Analysis
- Program Comprehension
- Requirements tracing
Traceability in practice

Traceability information is still not commonplace in software projects!
IR-based Traceability Recovery

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
<tr>
<td><strong>Events</strong></td>
<td></td>
</tr>
<tr>
<td>1.</td>
<td>The user opens the Laboratory GUI</td>
</tr>
<tr>
<td>2.</td>
<td>The user inserts the laboratory data</td>
</tr>
</tbody>
</table>
IR-based Traceability Recovery

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
<tr>
<td><strong>Events</strong></td>
<td></td>
</tr>
<tr>
<td>1:</td>
<td>The user opens the Laboratory GUI</td>
</tr>
<tr>
<td>2:</td>
<td>The user inserts the laboratory data</td>
</tr>
</tbody>
</table>

GUILaboratoryData.java

```java
/* *This class implements the GUI for managing laboratories data */

public class GUI Laboratory Data {
    private JFrame window;
    private JButton insert;
    ...

    public GUI Laboratory Data(){
        window = new JFrame();
        insert = new JButton();
        ...
    }
    ...
}
```
IR-based Traceability Recovery

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
<tr>
<td>Events</td>
<td>1: The user opens the Laboratory GUI</td>
</tr>
<tr>
<td></td>
<td>2: The user inserts the laboratory data</td>
</tr>
</tbody>
</table>

```java
public class GUILaboratoryData {
    /* *This class implements the GUI for managing laboratories data */
    public GUI Laboratory Data() {
        window = new JFrame();
        insert = new JButton();
        ...
    }
    ...
}
```
IR-based Traceability Recovery

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user opens the Laboratory GUI</td>
</tr>
<tr>
<td>2. The user inserts the laboratory data</td>
</tr>
</tbody>
</table>

GUILaboratoryData.java

```java
/* *This class implements the GUI for managing laboratories data */

public class GUILaboratoryData {
  private JFrame window;
  private JButton insert;
  ...

  public GUILaboratoryData() {
    window = new JFrame();
    insert = new JButton();
    ...
  }
}
```
IR-based Traceability Recovery

Source Artifacts ➔ IR Engine ➔ Target Artifacts

<table>
<thead>
<tr>
<th>Source</th>
<th>Target</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source_1</td>
<td>Target_2</td>
<td>95.4%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Target_4</td>
<td>92.1%</td>
</tr>
<tr>
<td>Source_1</td>
<td>Target_1</td>
<td>85.6%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Target_2</td>
<td>83.2%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Target_3</td>
<td>81.2%</td>
</tr>
<tr>
<td>Source_1</td>
<td>Target_3</td>
<td>79.0%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Target_2</td>
<td>77.5%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Target_4</td>
<td>64.3%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Target_3</td>
<td>53.2%</td>
</tr>
</tbody>
</table>
IR-based Traceability Recovery

Source Artifacts

IR Engine

Target Artifacts

Candidate Links

<table>
<thead>
<tr>
<th>Source</th>
<th>Target</th>
<th>Link Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source_1</td>
<td>Target_2</td>
<td>95.4%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Target_4</td>
<td>92.1%</td>
</tr>
<tr>
<td>Source_1</td>
<td>Target_1</td>
<td>85.6%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Target_2</td>
<td>83.2%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Target_3</td>
<td>81.2%</td>
</tr>
<tr>
<td>Source_1</td>
<td>Target_3</td>
<td>79.0%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Target_2</td>
<td>77.5%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Target_4</td>
<td>64.3%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Target_3</td>
<td>53.2%</td>
</tr>
</tbody>
</table>
Abstract

Existing methods for recovering traceability links among software documentation artifacts analyze textual documentation, whereas the problem of related documentation artifacts is not well studied or characterized. This paper presents a technique for identifying traceability links in requirements documentation by comparing textual with structural information. We experimentally show that related requirements share related source code elements. We also show that our approach improves the precision and recall of traceability links among requirements and documentation as compared to state-of-the-art text-based methods.

1. Introduction

Recovering traceability links requires not only the analysis of textual information derived from software artifacts, but also analysis of documentation. For example, if a developer writes code that is also in a related documentation artifact, then the code is likely to be mentioned in that documentation. However, a developer may not explicitly mention code in documentation, making text-based methods ineffective at discovering traceability links. Thus, we introduce a new method for recovering traceability links among source code and documentation. Our approach is based on finding related source code artifacts and their documentation artifacts.

Combining Textual and Structural Analysis of Software Artifacts for Traceability Link Recovery - Collin McMillan, Denys Poshyvanyk, Meghan Revelle

TEFSE 2009
Structural Information and Traceability

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
</tbody>
</table>
| Events         | 1. The user opens the Laboratory GUI  
                 2. The user inserts the laboratory data  
                 ...

public class GUILaboratoryData {
    JFrame window;
    JButton insert;
    ...
}

public GUILaboratoryData {
    window = new JFrame();
    insert = new JButton();
    ...
}

Similarity = 42%

/* *This class implements the GUI for managing laboratories data */
Structural Information and Traceability

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
<tr>
<td>Events</td>
<td>1. The user opens the Laboratory GUI</td>
</tr>
<tr>
<td></td>
<td>2. The user inserts the laboratory data</td>
</tr>
</tbody>
</table>

```java
public class Laboratory{
    private String name;
    private String position;
    ...
    public void setName(String pName){
        this.name=pName;
    }
    ...
}
```

```java
public class GUILaboratoryData {
    private JFrame window;
    private JButton insert;
    ...
    public GUILaboratoryData(){
        window = new JFrame();
        insert = new JButton();
        ...
    }
    ...
}
```

GUILaboratoryData.java

/* *This class implements the GUI for managing laboratories data */

public class GUILaboratoryData {
    private JFrame window;
    private JButton insert;
    ...
    public GUILaboratoryData(){
        window = new JFrame();
        insert = new JButton();
        ...
    }
    ...
}

Similarity = 42%
Structural Information and Traceability

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
<tr>
<td><strong>Events</strong></td>
<td>1. The user opens the Laboratory GUI</td>
</tr>
<tr>
<td></td>
<td>2. The user inserts the laboratory data</td>
</tr>
</tbody>
</table>

GUILaboratoryData.java

```java
/* *This class implements the GUI for managing laboratories data */

public class GUILaboratoryData {
    private JFrame window;
    private JButton insert;
    ...
}
```

Laboratory.java

```java
public class Laboratory {
    private String name;
    private String position;
    ...
    public void setName(String pName) {
        this.name = pName;
    }
    ...
}
```

Similarity = 42%

Similarity = 5%
### Use Case: Insert Laboratory Data

**Description:** The user inserts the data of a specific laboratory

**Events:**

1. The user opens the Laboratory GUI
2. The user inserts the laboratory data
   ...
Structural Information and Traceability

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
<tr>
<td>Events</td>
<td>1. The user opens the Laboratory GUI</td>
</tr>
<tr>
<td></td>
<td>2. The user inserts the laboratory data</td>
</tr>
</tbody>
</table>

GUILaboratoryData.java

```java
/* *This class implements the GUI for managing laboratories data */

public class GUILaboratoryData {
    private JFrame window;
    private JButton insert;
    ...

    public GUILaboratoryData()
    {
        window = new JFrame();
        insert = new JButton();
        ...
    }
}
```

Structural dependence

Laboratory.java

```java
public class Laboratory{
    private String name;
    private String position;
    ...
    public void setName(String pName){
        this.name=pName;
    }
    ...
}
```

Transitivity

Similarity = 42%

Similarity = 5%
**Structural Information and Traceability**

---

**Use Case**

**Insert Laboratory Data**

**Description**
The user inserts the data of a specific laboratory

**Events**

1. The user opens the Laboratory GUI
2. The user inserts the laboratory data
   
---

**GUILaboratoryData.java**

```java
/* *This class implements the GUI for managing laboratories data */

public class GUILaboratoryData {
    private JFrame window;
    private JButton insert;
    ...

    public GUILaboratoryData()
    {
        window = new JFrame();
        insert = new JButton();
        ...
    }
    ...
}
```

---

**Laboratory.java**

```java
public class Laboratory {
    private String name;
    private String position;
    ...

    public void setName(String pName) {
        this.name = pName;
    }
    ...
}
```

---

**Similarity = 42%**

---

**Similarity = 5% + BONUS**

---

**Transitivity**

---

**Structural Dependency**
Open Issues

1) The choice of the bonus value is crucial
   • Different systems require different bonus
   • Different IR methods require different bonus
Open Issues

1) The choice of the bonus value is crucial
   - Different systems require different bonus
   - Different IR methods require different bonus

2) When applying the bonus?
   - The transitivity property does not always hold
**Example**

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
<tr>
<td><strong>Events</strong></td>
<td>1. The user opens the Laboratory GUI</td>
</tr>
<tr>
<td></td>
<td>2. The user inserts the laboratory data</td>
</tr>
</tbody>
</table>

```java
/* *This class implements the GUI for managing laboratories data */

public class GUIDoctorData {
    private JFrame window;
    private JButton insert;
    ...

    public GUIDoctorData(){
        window = new JFrame();
        insert = new JButton();
        ...
    }
}
```

Not linked
### Example

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
</tbody>
</table>
| **Events** | 1. The user opens the Laboratory GUI  
2. The user inserts the laboratory data |

---

```java
/* *This class implements the GUI for managing laboratories data */

public class GUILaboratoryData {
    private JFrame window;
    private JButton insert;
    ...

    public GUILaboratoryData() {
        window = new JFrame();
        insert = new JButton();
        ...
    }
    ...
}
```

**GUIDoctorData.java**

```java
public class GUILaboratoryData {  
    private JFrame window;  
    private JButton insert;  
    ...

    public GUILaboratoryData() {
        window = new JFrame();
        insert = new JButton();
        ...
    }
    ...
}
```

*Similarity = 42%*
Example

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
</tbody>
</table>
| **Events** | 1. The user opens the Laboratory GUI  
2. The user inserts the laboratory data |

```java
public class GUILaboratoryData {
    private JFrame window;
    private JButton insert;
    ...
}
```

```java
/* *This class implements the GUI for managing laboratories data */

public class GUILaboratoryData {
    private JFrame window;
    private JButton insert;
    ...
}
```

```java
public class Authorization {
    public void setAuthorization(Doctor pDoctor, Laboratory pLab) {
        ...
    }
    ...
}
```

**Autorization.java**

**GUIDoctorData.java**

**Similarity = 42%**
**Example**

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
<tr>
<td><strong>Events</strong></td>
<td>1. The user opens the Laboratory GUI</td>
</tr>
<tr>
<td></td>
<td>2. The user inserts the laboratory data</td>
</tr>
</tbody>
</table>

**GUILaboratoryData.java**

```java
/* *This class implements the GUI for managing laboratories data */

public class GUILaboratoryData {
    private JFrame window;
    private JButton insert;
    ...
    public GUILaboratoryData(){
        window = new JFrame();
        insert = new JButton();
        ...
    }
    ...
}
```

**LaboratoryAuthorization.java**

```java
public class LaboratoryAuthorization{
    public void setAuthorization(Doctor pDoctor){
        ...
    }
    ...
}
```

**Similarity**

- **GUILaboratoryData.java**: 42%
- **LaboratoryAuthorization.java**: 5%
### Example

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
<tr>
<td><strong>Events</strong></td>
<td></td>
</tr>
<tr>
<td>1.</td>
<td>The user opens the Laboratory GUI</td>
</tr>
<tr>
<td>2.</td>
<td>The user inserts the laboratory data</td>
</tr>
</tbody>
</table>

---

**GUILaboratoryData.java**

```java
/* *This class implements the GUI for managing laboratories data */

public class GUILaboratoryData {
    private JFrame window;
    private JButton insert;
    ...

    public GUILaboratoryData()
    {
        window = new JFrame();
        insert = new JButton();
        ...
    }
}
```

**LaboratoryAuthorization.java**

```java
public class LaboratoryAuthorization{
    public void setAuthorization(Doctor pDoctor){
        ...
    }
    ...
}
```

**Similarity = 42%**

**Not linked**

---

**Similarity = 5% + BONUS**

---

**Structural dependency**

---

**Not linked**
Example

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Insert Laboratory Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The user inserts the data of a specific laboratory</td>
</tr>
<tr>
<td>Events</td>
<td>1. The user opens the Laboratory GUI</td>
</tr>
<tr>
<td></td>
<td>2. The user inserts the laboratory data</td>
</tr>
</tbody>
</table>

**GUILaboratoryData.java**

```java
public class GUILaboratoryData {
    JFrame window;
    JButton insert;
    ...
    public GUILaboratoryData() {
        window = new JFrame();
        insert = new JButton();
        ...
    }
    ...
}
```

**Laboratory.java**

```java
public class LaboratoryAuthorization{
    public void setAuthorization(Doctor pDoctor){
        ...
    }
    ...
}
```

**Use Case Insert Laboratory Data**

**Description**
The user inserts the data of a specific laboratory

**Events**
1. The user opens the Laboratory GUI
2. The user inserts the laboratory data

---

*Structural dependency*

*Structural information is not always useful*
**Approach 1:** Optimistic Combination (O-CSTI)

**Step 1: traditional IR process**

**Candidate Links**

<table>
<thead>
<tr>
<th>Source</th>
<th>Class</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source_1</td>
<td>Class_2</td>
<td>95.4%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_4</td>
<td>92.1%</td>
</tr>
<tr>
<td>Source_1</td>
<td>Class_1</td>
<td>85.6%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_2</td>
<td>83.2%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_3</td>
<td>81.2%</td>
</tr>
<tr>
<td>Source_1</td>
<td>Class_3</td>
<td>79.0%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_2</td>
<td>77.5%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_4</td>
<td>64.3%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_3</td>
<td>53.2%</td>
</tr>
</tbody>
</table>
**Approach 1: Optimistic Combination (O-CSTI)**

**Step 2: applying bonus to all the links**

### Candidate Links

<table>
<thead>
<tr>
<th>Source</th>
<th>Class</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source_1</td>
<td>Class_1</td>
<td>85.6%</td>
</tr>
<tr>
<td>Source_1</td>
<td>Class_2</td>
<td>83.2%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_2</td>
<td>81.2%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_2</td>
<td>79.0%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_3</td>
<td>77.5%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_4</td>
<td>64.3%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_4</td>
<td>53.2%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_4</td>
<td>92.1%</td>
</tr>
</tbody>
</table>

+ Bonus

+ Bonus
**Approach 1:** Optimistic Combination (O-CSTI)

**Step 2:** applying bonus to all the links

![Diagram showing candidate links]

<table>
<thead>
<tr>
<th>Source</th>
<th>Class</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source_1</td>
<td>Class_2</td>
<td>95.4%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_4</td>
<td>92.1%</td>
</tr>
<tr>
<td>Source_1</td>
<td>Class_1</td>
<td>85.6%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_2</td>
<td>83.2%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_3</td>
<td>81.2%</td>
</tr>
<tr>
<td>Source_1</td>
<td>Class_3</td>
<td>79.0%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_2</td>
<td>77.5%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_4</td>
<td>64.3%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_4</td>
<td>53.2%</td>
</tr>
</tbody>
</table>

+ Bonus  
+ Bonus  
+ Bonus
**Approach 1: Optimistic Combination (O-CSTI)**

**Step 2: applying bonus to all the links**

![Diagram of candidate links with classes and source links]

<table>
<thead>
<tr>
<th>Candidate Links</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Source_1</td>
<td>Class_2</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_4</td>
</tr>
<tr>
<td><strong>Source_1</strong></td>
<td>Class_1</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_2</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_3</td>
</tr>
<tr>
<td>Source_1</td>
<td>Class_3</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_2</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_4</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_4</td>
</tr>
</tbody>
</table>

+ Bonus
+ Bonus
+ 2Bonus
+ Bonus
Approach 1: Optimistic Combination (O-CSTI)

1. Traditional IR process
2. Apply the bonus to all the links
3. Re-order the list
4. The user judges the i-th link

End?

NO

YES
**Approach 2:** User Driven Combination (U-CSTI)

*Only the user can say if a link is *correct* or *not***
Approach 2: User Driven Combination (U-CSTI)

Only the user can say if a link is correct or not

Only the user can say if applying the combination or not
Approach 2: User Driven Combination (U-CSTI)

1. Traditional IR process
2. The user judges the $i$-th link
   - if correct
     - Apply bonus
     - Re-order the list
   - else
     - No bonus
3. End?
   - YES
   - NO
Adaptive bonus

$$Sim(Source1, \text{Class1}) = Sim(Source1, \text{Class1}) + \delta \ast Sim(Source1, \text{Class1})$$
Adaptive bonus

\[ \text{Sim(Source1, Class1)} = \text{Sim(Source1, Class1)} + \delta \times \text{Sim(Source1, Class1)} \]

\[ \delta = \frac{\max(\text{Sim}) - \min(\text{Sim})}{2} \approx \text{ranked list variability} \]

Candidate Links

<table>
<thead>
<tr>
<th>Source</th>
<th>Class</th>
<th>Sim (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source_1</td>
<td>Class_2</td>
<td>95.4%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_4</td>
<td>92.1%</td>
</tr>
<tr>
<td>Source_1</td>
<td>Class_1</td>
<td>85.6%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_2</td>
<td>83.2%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_3</td>
<td>81.2%</td>
</tr>
<tr>
<td>Source_1</td>
<td>Class_3</td>
<td>79.0%</td>
</tr>
<tr>
<td>Source_3</td>
<td>Class_2</td>
<td>77.5%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_4</td>
<td>64.3%</td>
</tr>
<tr>
<td>Source_2</td>
<td>Class_3</td>
<td>53.2%</td>
</tr>
</tbody>
</table>
Implementation

TraceLab Components

1) Adaptive Bonus
2) Optimistic Combination (O-CSTI)
3) User Driven Combination (U-CSTI)
4) Different IR Methods:
   - Vector Space Model
   - Jensen-Shannon

We provide the experiments and datasets for download at http://www.cs.wm.edu/semuru/data/csmr13/
Implementation

We provide the experiments and datasets for download at http://www.cs.wm.edu/semeru/data/csmr13/

TraceLab Components

1) Adaptive Bonus
2) Optimistic Combination (O-CSTI)
3) User Driven Combination (U-CSTI)
4) Different IR Methods:
   - Vector Space Model
   - Jensen-Shannon
Empirical Evaluation
We compared three IR-based processes:
1) IR process alone
2) O-CSTI (optimistic combination)
3) U-CSTI (user driven combination)
Results

Vector Space Model

![Bar chart showing results for UC-CC, ID-CC, TC-CC, UC-CC EasyClinic, UC-CC eTour, UC-CC SMOS using Vector Space Model with IR alone, U-CSTI, and O-CSTI. The chart displays average precision values.]
Results

Jensen-Shannon Divergence

![Bar chart showing average precision for different categories and models.](chart.png)
Results

Tracing Use Cases onto Code Classes on SMOS

Vector Space Model
Results

Tracing Use Cases onto Code Classes on SMOS

Vector Space Model

Jensen-Shannon Divergence

- IR alone
- UD-CSTI
- O-CSTI
Optimality of the Adaptive Bonus

Adaptive Bonus vs Fixed Bonus
- We used different fixed bonus values
Optimality of the Adaptive Bonus

Adaptive Bonus vs Fixed Bonus
- We used different fixed bonus values

EasyClinic UC-CC with VSM

EasyClinic TC-CC with JS
Conclusions

Approach 1: Optimistic Combination (O-CSTI)

Approach 2: User Driven Combination (UC-CSTI)

Adaptive bonus

\[
\text{Sim}(\text{Source}, \text{Class}) = \text{Sim}(\text{Source}, \text{Class}) + \alpha \times \text{Sim}(\text{Source}, \text{Class})
\]

Results

Optimality of the Adaptive Bonus
THANKS