Haijuan WANG, Guohua SHEN, Zhiqiu HUANG, Yaoshen YU, Kai CHEN, 2021. Analyzing close relations between target artifacts for improving IR-based requirement traceability recovery. *Frontiers of Information Technology & Electronic Engineering,* 22(7):957-968. https://doi.org/10.1631/FITEE.2000126



# Analyzing close relations between target artifacts for improving IR-based requirement traceability recovery

**Key words:** Requirement traceability; Information retrieval; Close relations; Target artifacts

Corresponding author: Guohua SHEN
E-mail: ghshen@nuaa.edu.cn
ORCID: https://orcid.org/0000-0003-2182-0019

# Motivation

1. Requirement traceability is an important and costly task.

2. The information retrieval (IR) technique has been widely used in requirement traceability. However, if two artifacts do not share or share only a small number of words, the performance of IR can be very poor.

3. To overcome this limitation, we propose an automatic method that combines the IR method with the close relations between target artifacts.

# Main idea

1. We propose an automatic method that combines the IR method with the close relations between target artifacts (called IR_CRT).

2. Specifically, we leverage close relations between target artifacts rather than just text matching from requirements to target artifacts. Moreover, the method is not limited to the type of target artifacts when considering the relations between target artifacts.

# Method

1. The IR_CRT approach includes mainly the following steps:
   - First, a traditional IR-based traceability recovery method is used to generate an original candidate list for each requirement.
   - Second, we need to cut the ranked candidate list to form the initial requirement region for each requirement. For each target artifact, through word embedding, we can obtain the close relation list ranking from high to low in the similarity value. Then, we also need to cut the ranked candidate list to form the CRTG.
   - Third, for each requirement, inside-region target artifacts are directly added to a new candidate list, and the outside-region target artifacts are re-ranked (Algorithm 1) by considering whether to increase a bonus. Then we add the candidate trace links with the recalculated similarity value to the new candidate list.

# Method (Cont'd)

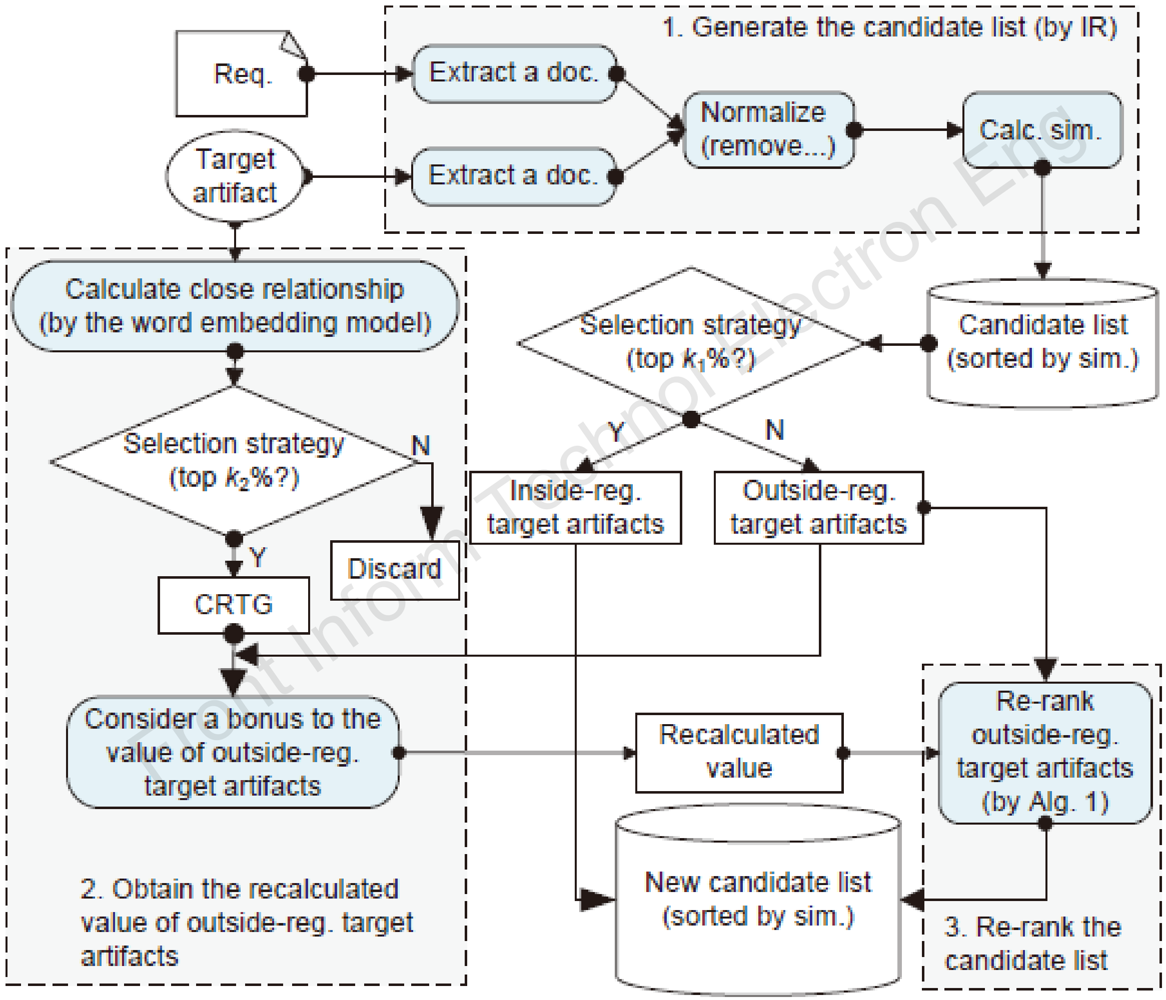


**Fig. 3 Candidate list generation and re-ranking process**

# Method (Cont'd)

2. Datasets for the experiments

(we used five datasets from five software projects)

**Table 1 System information**

| System | Source artifact (number of source artifacts) | Target artifact (number of target artifacts) | Number of correct links |
|---|---|---|---|
| EasyClinic | UC (30) | TC (63) | 63 |
| CM1-NASA | Req. (235) | Design (220) | 353 |
| Pine | Req. (49) | UC (51) | 250 |
| GANNT | HR (17) | LR (69) | 68 |
| iTrust | UC (36) | CC (106) | 174 |

UC: use case; TC: test case; HR: high-level requirements; LR: low-level requirements; CC: class code

# Method (Cont'd)

3. Selection of two cut percentages and bonus values

- Original requirement region
- CRTG
- An adaptive bonus that is proportional to the median variability of the similarity values computed for each software artifact.

# Method (Cont'd)

4. Re-ranking

**Algorithm 1** Re-ranking the original candidate list

**Input:** Original candidate list

**Output:** Re-ranked candidate list

$i \leftarrow 1$ // $i$ means the position of a trace link in the // candidate list
**while** not (end of List) **do**
  Obtain the top $k\%$ links $(s,\ t_j)$ in position $i$ of List
  **for** all $t_p \in T$ **do**
    **if** $(t_j,\ t_p) \in E$ **then**
      $\mathrm{Sim}(s,\ t_p) \leftarrow \mathrm{Sim}(s,\ t_p) + \delta \cdot \mathrm{Sim}(s,\ t_p)$ // Each edge $(t_i,\ t_j) \in E$ represents a close // relation
    **end if**
  **end for**
  $i \leftarrow i + 1$
**end while**
Re-ranked List

# Method (Cont'd)

5. Two research questions:

- **RQ1:** What is the best setting for our method (such as the size of the initial requirement region)?
- **RQ2:** Whether our method is generic to create links from requirements to *x* (*x* can be test cases, designs, code, etc.) without considering what type of artifact the target is?

# Major results

- To answer RQ1: determining the best setting of IR_CRT method

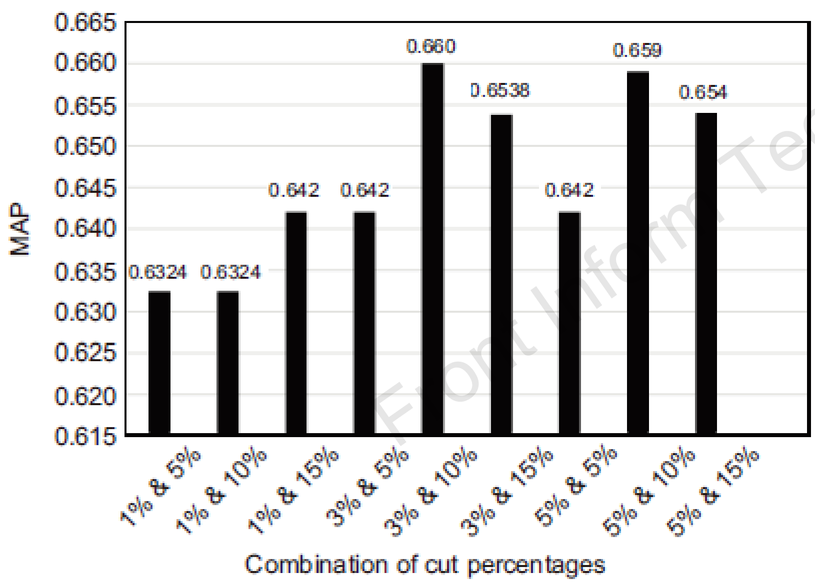


**Fig. 6** MAP values of IR_CRT at different cut percentage values on EasyClinic

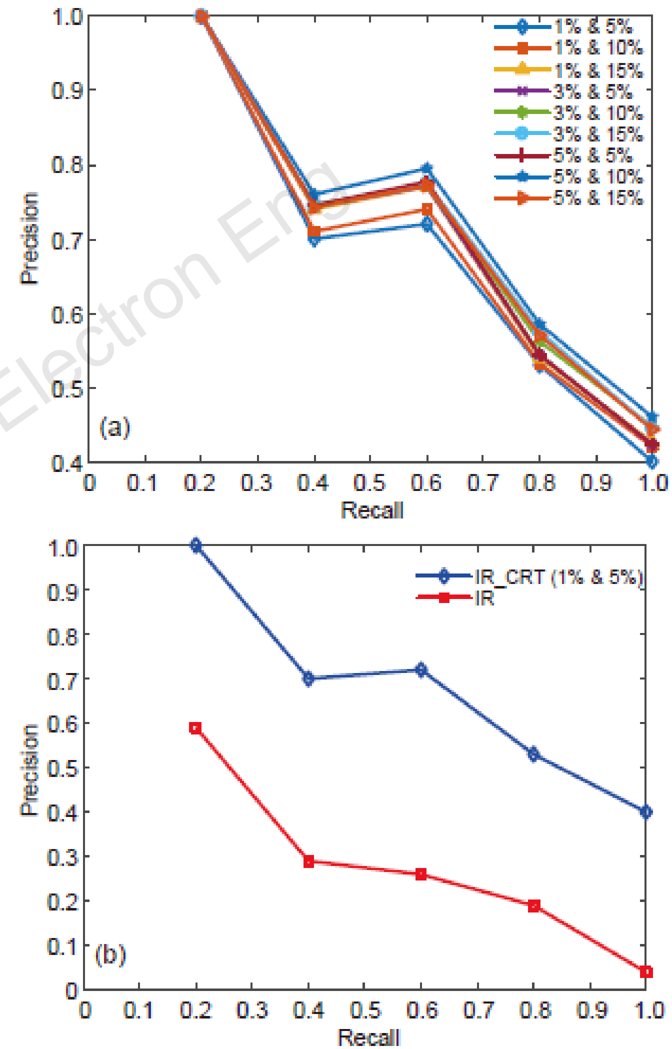


**Fig. 5** Precision–recall curves on EasyClinic: (a) IR_CRT at different cut percentage values; (b) IR_CRT (1% & 5%) versus IR

# Major results (Cont'd)

- To answer RQ2: verifying that the IR_CRT method is not limited to the target artifact type



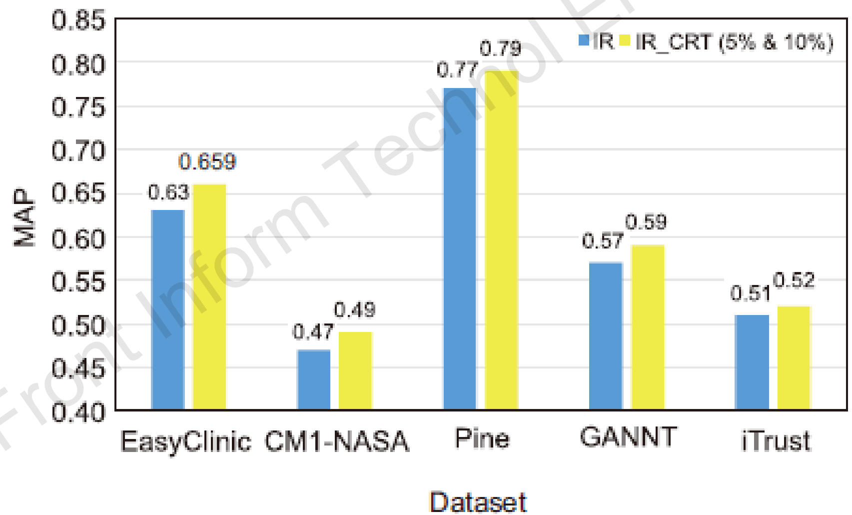


**Fig. 7** MAP values of IR versus IR_CRT (5% & 10%) on different datasets

# Major results (Cont'd)

- To answer RQ2: verifying that the IR_CRT method is not limited to the target artifact type

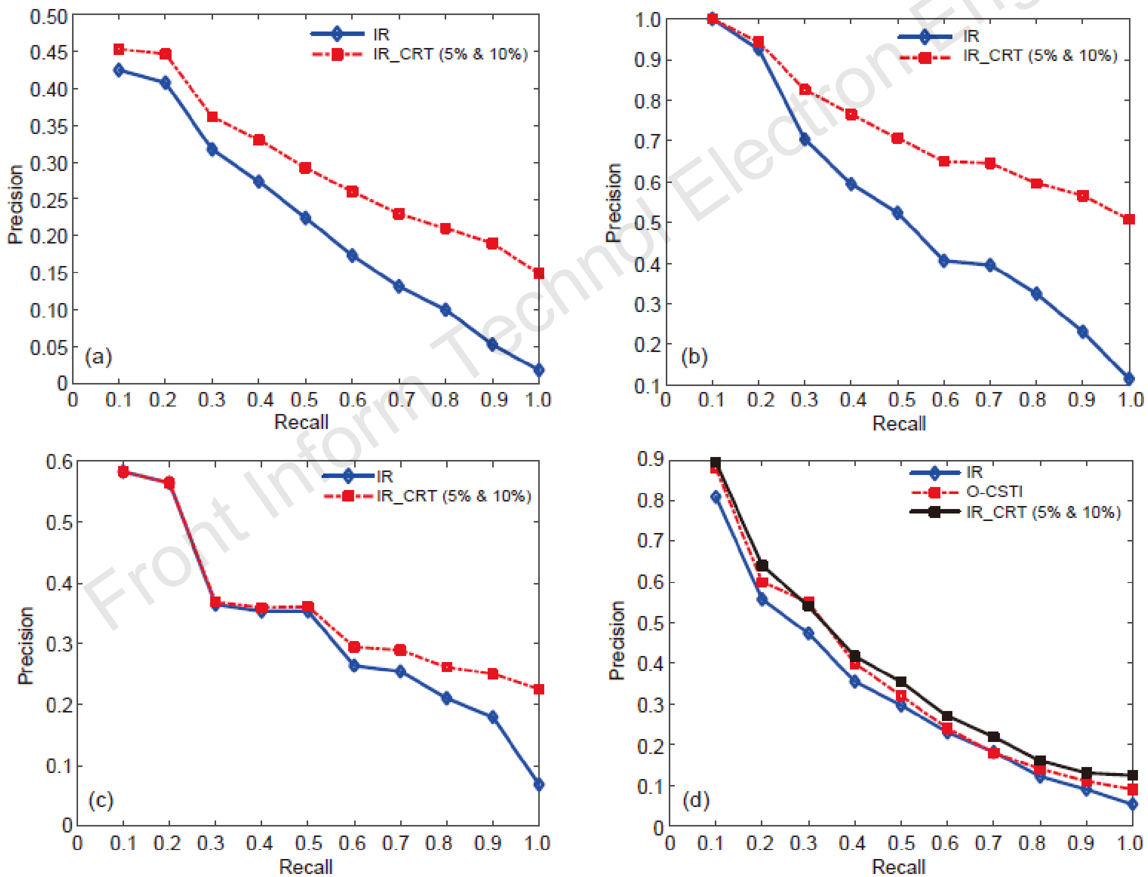



Fig. 8 Comparison of the precision–recall curves of IR versus IR_CRT (5% & 10%): (a) CM1-NASA; (b) Pine; (c) GANNT; (d) iTrust (since the type of its target artifact is code, we also add the O-CSTI result for comparison)

# Major results (Cont'd)

- To answer RQ2: verifying that the IR_CRT method is not limited to the target artifact type

**Table 2 An example of the re-ranking part of the target artifacts of $UC_7$ (EasyClinic)**

| Source artifact | Target artifact | Similarity score (initial) | Ranking (initial) | Similarity score (recalculated) | Re-ranking | isTrace |
|---|---|---|---|---|---|---|
| $UC_7$ | $TC_{51}$ | 0.320 63 | 1 | 0.423 433 596 9 | 1 | × |
| $UC_7$ | $TC_{56}$ | 0.289 95 | 2 | 0.382 916 668 5 | 2 | × |
| $UC_7$ | $TC_{55}$ | 0.286 59 | 3 | 0.378 479 351 7 | 3 | × |
| $UC_7$ | $TC_{54}$ | 0.282 88 | 4 | 0.373 579 814 4 | 4 | × |
| $UC_7$ | $TC_{52}$ | 0.279 67 | 5 | 0.369 340 592 1 | 5 | × |
| $UC_7$ | $TC_{58}$ | 0.273 86 | 6 | 0.273 86 | 8 ↓ | |
| $UC_7$ | $TC_{62}$ | 0.273 27 | 7 | 0.273 27 | 9 ↓ | |
| $UC_7$ | $TC_{53}$ | 0.271 60 | 8 | 0.358 683 108 | 6 ↑ | × |
| $UC_7$ | $TC_{57}$ | 0.232 51 | 9 | 0.307 059 681 3 | 7 ↑ | × |
| $UC_7$ | $TC_{60}$ | 0.041 22 | 10 | 0.053 586 | 10 | |

UC: use case; TC: test case; ×: trace link

# Conclusions

1. An empirical evaluation conducted on five software systems suggests that our approach outperforms the baseline method.

2. We choose an appropriate cut percentage combination (5% and 10%).

3. The precision of the five datasets is improved by an average of 15.6%.