Recovering Traceability Links in Requirements Documents

Zeheng Li, Mingrui Chen, LiGuo Huang, and Vincent Ng

1Southern Methodist University 2The University of Texas at Dallas

Task

- Given a set of high-level requirements and a set of low-level requirements, recover the traceability links between them
- Two requirements should be linked if one is a refinement of the other

Requirements traceability is many-to-many mapping:
- A high-level requirement can be refined by multiple low-level requirements
- A low-level requirement can refine multiple high-level requirements

HR03 is refined by UC02

Abundant information irrelevant to the link establishment

Software system development is guided by the evolution and refinement of many-to-many requirements.

Why is it important for Software Engineering?

- Software system development is guided by the evolution and refinement of requirements
- Requirements specifications are refined with additional design details and implementation information as the development lifecycle progresses

Why is it challenging for NLP?

- Abundant information irrelevant to the link establishment
- Information irrelevant to the establishment of one link could be relevant to the establishment of another link involving the same requirement

Datasets

- Pine - Email system of University of Washington
- WorldVistA – Health information system

Baseline Systems

I. Unsupervised baselines

- Link two documents if their Cosine similarity exceeds a certain threshold
- Employ two ways to represent a document as a vector of unigrams
  - Feature values are the tf-idf values
  - as a vector of n topics induced by an LDA model
    - Feature values are the probabilities the document belongs to the topics
    - n = 10, 20, … 50 (Pine) and 50, 60, … 100 (WorldVistA)
    - n is tuned on test data (thus giving an unfair advantage to these baselines)

II. Supervised baseline

- Linking decisions made by a binary classifier trained using LibSVM
  - Create instances by pairing each high-level requirement with each low-level requirement
  - Positive if the two requirements should be linked, and negative otherwise
  - Two types of binary-valued features:
    - Word pairs: a pair of words \( (w_i, w_j) \) from the high- and low-level documents respectively, indicating their presence in these documents
    - LDA-induced topic pairs: topic pair \( (t_i, t_j) \) whose values is 1 if \( t_i \) and \( t_j \) are the most probable topics for the high- and low-level requirements
  - C (the regularization parameter) is tuned on development data

Knowledge-rich Approach

Goal: Improve supervised baseline using two types of human-supplied knowledge

1. Noun and verb clusters

- Two ways to create noun and verb clusters
  - Manually
    - First define domain-relevant noun and verb categories, then populate them
    - Pine: 8 noun clusters and 10 verb clusters
    - WordVistA: 31 noun clusters and 14 verb clusters
    - A time-consuming process
  - Automatically (using single-link agglomerative clustering)
    - Each noun (verb) is represented using the verbs (nouns) it co-occurs with
    - We only cluster nouns/verbs in the training data that (1) have at least three characters, and (2) appear in only high-level or only low-level documents
    - Each noun/verb is initially in its own cluster
    - In each iteration, it merges the two most similar clusters and stops when the desired number of clusters is reached
    - Number of clusters: 10, 15, 20 (Pine) and 10, 20, 30, 40, 50 (WorldVistA)

- Use the manual/induced word clusters to create additional types of features
  - Verb pairs: pairs of verbs collected from high- and low-level requirements
  - Noun groups pairs: pairs of nouns collected from high- and low-level requirements
  - Noun group pairs: replace nouns in the high-level document with nouns in the low-level document
  - Dependency pairs: created by pairing each noun-verb pair found in high-level requirement with each noun-verb pair found in low-level requirement
  - Cluster-based features can provide better generalization than word-based features

II. Annotator rationales

- Rationales are words/phrases in a training document that are considered relevant to the classification task at hand by the human annotator
- For each link in the training set, we asked the annotator to identify words/phrases from the associated requirements that are relevant to establishing the link
- Rationales are used to create two types of additional training instances
  - Three pseudo negative instances are created from each negative training instance
  - The first is created by removing only rationales from high-level document
  - The second is created by removing only rationales from low-level document
  - The third is created by removing rationales from both requirements
  - Potentially allow the learner to focus on learning from the relevant phrases

Example

A1: The system shall have an address book available to store contacts.
- Terms in red are rationales that are helpful for recovering the link
- Terms in blue results in pseudo negative instance

Evaluation

- Evaluation metrics
  - Recall: percentage of recovered links in the gold standard
  - Precision: percentage of correctly recovered links
  - F-score: unweighted harmonic mean of recall and precision

Results

<table>
<thead>
<tr>
<th>System</th>
<th>Pine</th>
<th>WorldVistA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No Pseudo</td>
<td>Pseudo pos+neg</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>P</td>
</tr>
<tr>
<td>Tf-idf Baseline</td>
<td>73.6</td>
<td>43.3</td>
</tr>
<tr>
<td>LDA Baseline</td>
<td>56.0</td>
<td>39.2</td>
</tr>
<tr>
<td>Supervised baseline + manual clusters</td>
<td>50.4</td>
<td>67.0</td>
</tr>
<tr>
<td>Supervised baseline + induced clusters</td>
<td>54.9</td>
<td>73.9</td>
</tr>
</tbody>
</table>
| Discussion    | When pseudo-instances are not used,
  - the Tf-idf baseline significantly outperforms the LDA baseline
  - the supervised baseline significantly outperforms the two unsupervised baselines
  - adding cluster-based features significantly improve the results of the supervised baseline
  - when pseudo-instances are used,
  - adding cluster-based features significantly improve the results of the supervised baseline
  - results are significantly better than when no pseudo-instances are used
  - Relative error reductions of 11.1-19.7% compared to the tf-idf baseline