1. Our goal
- Current goal: Help to have a better understanding of the rewriting process
- Ultimate goal: Provide automatic revision suggestions

2. Why do we need sentence-level rewriting detection?

- Revision detection at word-level
  - Typical word-level approach cannot accurately reflect the essence of the document
  - Like, if we want to know how two sentences change from the first draft to the second one, we need to find the exact locations of changes and their types
  - However, this information is usually not available

- Current word-level approach can only find the locations of changes
  - Diff alignment method compares the insertions and red for deletion

- Our work
  - Sentence alignment: align sentences of the revised document to the ones of the original document
    - Allows many-to-one and one-to-many alignments
    - For the example above:
      - Line 55 aligned to line 56
      - Line 57 and 58 are aligned to line 59

- Edit sequence generation: generate the edit sequence from the original document to the revised
  - 4 basic primitives: Add, Delete, Modify, Keep
  - For the example above:
    - Sentence Index (First): 54, 55, 56, 57
    - Sentence Index (Second): 54, 55, 56, 57

- Edit sequence merging: merge the basic primitives into more meaningful advanced edit primitives
  - Advanced edit primitives: permutation, distribution, consolidation
  - For the example above:
    - Sentence Index (First) 54, 55, 56, 57
    - Sentence Index (Second) 54, 55, 56, 57

3. Our work
- Data preparation
  - 2 undergraduate paper assignments from a “Social Implications of Computing Technology” course
  - Collected via a web-based peer review system[1], each paper has two drafts

- Manual annotation
  - Sentence alignment: two annotators annotate on one paper, kappa: 0.794
  - Edit sequence generation: annotate edit sequence from the first draft
  - Edit sequence merging: annotates “consolidation”, “permutation” currently

- Automatic sentence-level revision detection in 3 steps
  - Sentence alignment
    - Method: adapting Nenel’s approach[3]
      - Logistic regression classifier using sentence similarity score (Word Overlap, TF-IDF, Levenshtein Distance)
    - Global alignment based on sentence order (Needleman-Wunsch[4])
    - Evaluation: accuracy (percentage of sentences that are correctly aligned)

- Performance
  - Baseline: Hashemi’s word-based approach (as in section 1), performance collected by manual inspection
  - Rule-based approach used in edit sequence generation phase with approach based on edit distance, and then infer advanced edits based on the automatic generated sequence

- Edit sequence generation
  - Method: Rule-based approach
  - Evaluation: Word error rate (WER), rate of segments to be modified to match with the correct sequence
  - Performance
    - Baseline: Hashemi’s word-based approach (as in section 1), performance collected by manual inspection
    - Rule-based method
      - Rate of segments to be modified to match with the correct sequence

- Edit sequence merging
  - Method: Rule-based approach, now only recognizes “Distribution” and “Consolidation”
  - Evaluation: accuracy (percentage of the “Distribution” and “Consolidation” cases recognized)
  - Performance: The 9 consolidation and 5 distribution cases are all successfully identified

4. Future work
- Improve the accuracy of current algorithm
- Replace rule-based approach used in edit sequence generation phase with approach based on edit distance, and then infer advanced edits based on the automatic generated sequence
- Identify more meaningful advanced rewriting operations
- Conduct user study comparing the utility of sentence versus word-level rewriting detection

References

This research is supported by the Institute of Educational Sciences, U.S. Department of Education, through Grant R305A120370 to the University of Pittsburgh. The opinions expressed are those of the authors and do not necessarily represent the views of the Institute or the U.S. Department of Education.