

Jointly Preprocessed errant

Evaluation for End-to-End GEC

Refined Evaluation for End-to-End Grammatical Error Correction Using an Alignment-Based Approach

Junrui Wang¹ Mengyang Qiu^{2,3} Yang Gu³ Zihao Huang³ Jungyeul Park^{1,3}

¹The University of British Columbia ²Trent University ³Open Writing Evaluation

Procedure example of jp-errant:

1. Preparation	
gold m2	S Kate Ashby , A -1 -1 noop NONE- REQUIRED NONE- 0 S how are you ? I hope you are well . A 0 1 R:ADV How REQUIRED NONE- 0
stanza m2	S Kate Ashby , how are you ? A -1 -1 noop NONE- REQUIRED NONE- 0 S I hope you are well . A -1 -1 noop NONE- REQUIRED NONE- 0
2. Sentence alignment	
gold m2	S Kate Ashby , how are you ? I hope you are well . A -1 -1 noop NONE- REQUIRED NONE- 0 A 0 1 R:ADV How REQUIRED NONE- 0
stanza m2	S Kate Ashby , how are you ? I hope you are well . A -1 -1 noop NONE- REQUIRED NONE- 0 A -1 -1 noop NONE- REQUIRED NONE- 0
3. Re-indexing	
gold m2	S Kate Ashby , how are you ? I hope you are well . A 3 4 R:ADV How REQUIRED NONE- 0
stanza m2	S Kate Ashby , how are you ? I hope you are well . A -1 -1 noop NONE- REQUIRED NONE- 0

Differences between jp-errant and errant:

jp-errant	errant
S It 's difficult answer at the question " ... A 3 3 M:VERB:FORM to REQUIRED NONE- 0 A 4 5 U:PREP REQUIRED NONE- 0	S It 's difficult answer at the question " ... A 3 3 M:VERB:FORM to REQUIRED NONE- 0 A 4 5 U:ADP REQUIRED NONE- 0
S Thank you for your e - mail , it was wonderful to hear from you . A 3 4 R:PRON your REQUIRED NONE- 0 A 7 9 R:PUNCT . It REQUIRED NONE- 0	S Thank you for your e - mail , it was wonderful to hear from you . A 3 4 R:DET your REQUIRED NONE- 0 A 7 9 R:PUNCT . It REQUIRED NONE- 0

Algorithm

```
1: function PATTERNMATCHINGSA ( $\mathcal{L}$ ,  $\mathcal{R}$ ):
2:   while  $\mathcal{L}$  and  $\mathcal{R}$  do
3:     if  $\mathcal{L}_{i(\mathcal{L})} = \mathcal{R}_{j(\mathcal{R})}$  then
4:        $\mathcal{L}', \mathcal{R}' \leftarrow \mathcal{L}' + \mathcal{L}_i, \mathcal{R}' + \mathcal{R}_j$  where
          $0 < i \leq \text{LEN}(\mathcal{L}), 0 < j \leq \text{LEN}(\mathcal{R})$ 
5:     else
6:       while  $\neg(\mathcal{L}_{i(\mathcal{L})} = \mathcal{R}_{j(\mathcal{R})})$  do
7:         if  $\text{LEN}(\mathcal{L}_i) < \text{LEN}(\mathcal{R}_j)$  then
8:            $\mathcal{L}' \leftarrow \mathcal{L}' + \mathcal{L}_i$ 
9:            $i \leftarrow i + 1$ 
10:        else
11:           $\mathcal{R}' \leftarrow \mathcal{R}' + \mathcal{R}_j$ 
12:           $j \leftarrow j + 1$ 
13:        end if
14:      end while
15:       $\mathcal{L}', \mathcal{R}' \leftarrow \mathcal{L}' + \mathcal{L}', \mathcal{R}' + \mathcal{R}'$ 
16:    end if
17:  end while
18:  return  $\mathcal{L}', \mathcal{R}'$ 
```

The proposed alignment approach addresses inconsistencies caused by tokenization differences in sentence pairs. Sequences \mathcal{L}_i and \mathcal{R}_j are initially aligned by removing spaces to minimize tokenization-induced differences ($\mathcal{L}_i \setminus \text{ } == \mathcal{R}_j \setminus \text{ }$). Tokenization variations, such as contractions (e.g., *can't* tokenized as *ca n't* or *can not*), often require more nuanced methods.

Sequences are aligned if their character-level similarity exceeds a threshold α , and subsequent sequences ($\mathcal{L}_{i+1}, \mathcal{R}_{j+1}$) also meet similarity or matching criteria (Equation (??)). A modified Jaro-Winkler distance, incorporating prefix and suffix scales, calculates α :

$$\alpha = \text{sim}_j - \frac{(lp + l'p)(1 - \text{sim}_j)}{2} \quad (1)$$

where sim_j is the Jaro similarity between two strings s_1 and s_2 , l and l' are the lengths of the common prefix and suffix, and p is a scaling factor (set to 0.1).

