An Untold Story of Preprocessing Task Evaluation: An Alignment-based Joint Evaluation Approach

Eunkyul Leah Jo[†‡*] Angela Yoonseo Park[†*] Grace Zhang[†*]
Izia Xiaoxiao Wang[¶*] Junrui Wang[†*] MingJia Mao[†*]
Jungyeul Park[†]

[†]The University of British Columbia, Canada [‡]Sorbonne Université, France
[¶]Universität Zürich, Schweiz. [*]Equally contributed authors.

LREC-COLING 2024
20-25 May, 2024

1. We introduce a novel evaluation algorithm for the preprocessing task, including both tokenization and SBD results.
2. This algorithm aims to enhance the reliability of evaluations by reevaluating the counts of true positive cases for F1 measures in both preprocessing tasks jointly.
3. It achieves this through an alignment-based approach inspired by sentence and word alignments used in machine translation.
4. Our evaluation algorithm not only allows for precise counting of true positive tokens and sentence boundaries but also combines these two evaluation tasks into a single organized pipeline.

Contents

Mismatches in Preprocessing

Tokenization

`tok.sed`	When No␣. 1 is␣n't the Best
`Moses`	When No. 1 isn␣'t the Best
`CoreNLP`	When No. 1 is␣n't the Best

where ␣ is a symbol for a token delimiter.

Sentence boundary detection

```
Opening of the session
I declare resumed the
2000-2001 session of the
European Parliament.
```

Many common sentence boundary detection systems, such as `splitta` (Gillick, 2009), `CoreNLP` (Manning et al., 2014), and `Elephant` (Evang et al., 2013), often fail to correctly identify sentence boundaries in cases where there are no punctuation marks between two sentences

Evaluation

Methods for evaluating sentence boundary disambiguation and tokenization can be broadly categorized into three main approaches:

Accuracy and error rate
: This method assesses how accurately the boundaries are detected or tokens are separated, and quantifies the rate of errors made.

Precision and recall for F1 Measure
: This approach focuses on measuring the precision (how many of the detected boundaries or tokens are correct) and recall (how many of the actual boundaries or tokens are detected). The F1 measure is then calculated to balance these two factors.

Levenshtein distance
: This method involves calculating the Levenshtein distance, which represents the number of edits (insertions, deletions, or substitutions) needed to transform the detected boundaries or tokens into the correct ones.

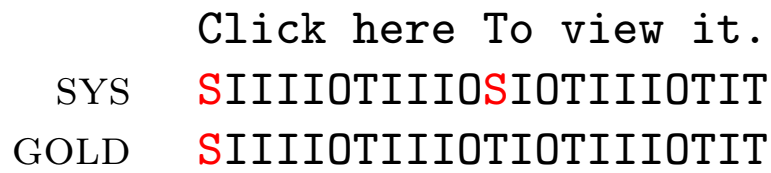

Figure 1: BIO-style evaluation in previous work where partial matched sentences (e.g *Click here*) could be considered as `true positive`.

The first sentence boundary marked by *Click* is essentially a partial match, while the second part is separated into another sentence by the second sentence boundary marked by *To*.

Alignment-based Evaluation

input file:
`Click here To view it.  He makes some good ⊓`
`observations on a few of the picture's. ⊓`

system result:
`Click here ⊓`
`To view it␣.  ⊓`
`He makes some good observations on a few of the picture␣'s␣.  ⊓`

gold file:
`Click here To view it␣.  ⊓`
`He makes some good observations on a few of the picture␣'s␣.  ⊓`

sentence-aligned result for preprocessing evaluation:
`Click here ∼∼∼ To view it␣.  ⊓` `Click here To view it␣.  ⊓`
`He makes some good observations` `He makes some good observations`
`on a few of the picture␣'s␣.  ⊓` `on a few of the picture␣'s␣.  ⊓`

Figure 2: Example of intermediate results of the evaluation by alignment algorithm where *Click here* and *To view it.* In the system result are the realigned, produced by merging after the implementation of sentence alignment: ␣ is a symbol for tokenization, ⊓ is a symbol for SBD, and ∼∼∼ is a symbol for merged sentences by sentence alignment.

Evaluation measures

$$\begin{aligned}
\text{precision} &= \frac{\text{relevant \# of } \texttt{sb} \cap \text{retrieved \# of } \texttt{sb}}{\text{retrieved \# of } \texttt{sb}} \\
&= \frac{\mathcal{C}_{sb}(\mathcal{L}) \cap \mathcal{C}_{sb}(\mathcal{R})}{\mathcal{C}_{sb}(\mathcal{L})} = \frac{\texttt{TP}_{\texttt{sb}}}{\mathcal{C}_{sb}(\mathcal{L})} \\
\text{recall} &= \frac{\text{relevant \# of } \texttt{sb} \cap \text{retrieved \# of } \texttt{sb}}{\text{relevant \# of } \texttt{sb}} \\
&= \frac{\mathcal{C}_{sb}(\mathcal{L}) \cap \mathcal{C}_{sb}(\mathcal{R})}{\mathcal{C}_{sb}(\mathcal{R})} = \frac{\texttt{TP}_{\texttt{sb}}}{\mathcal{C}_{sb}(\mathcal{R})}
\end{aligned}$$

To apply in the algorithm, 'relevant # of `sb` ∩ retrieved # of `sb`' represents $\texttt{TP}_{\texttt{sb}}$, 'retrieved # of `sb`' is $\mathcal{C}_{sb}(\mathcal{L})$ by a system result, and 'relevant # of `sb`' is $\mathcal{C}_{sb}(\mathcal{R})$ by a gold file. The same precision and recall can be defined for tokenization using $\mathcal{C}_{tk}(\mathcal{L})$, $\mathcal{C}_{tk}(\mathcal{R})$ and $\texttt{TP}_{\texttt{tk}}$.

Notations

$\mathcal{C}_{sb}(\mathcal{L})$ and $\mathcal{C}_{tk}(\mathcal{L})$: These represent the total number of sentence boundaries (sb) and the total number of tokens (tk) in $\mathcal{L}$.

$\texttt{TP}_{\texttt{sb}}$ and $\texttt{TP}_{\texttt{tk}}$: These denote the number of true positives (`tp`) for sentence boundaries and tokens, respectively.

$L_{\sqcup}$: This represents L where spaces between tokens have not been removed.

$L_{\not\sqcup}$: This represents L where spaces between tokens have been removed.

L_i: It stands for the ith token in $L_{\sqcup}$.

`jp-algorithm` for preprocessing evaluation

Algorithm 1 Pseudo-code for alignment

```
function ALIGNMENT (L, R):
    while L and R do
        if Matched CASES (i,j) then
            L', R' ← L' + L_i, R' + R_j
        else
            while ¬(Matched CASES (i+1,j+1) do
                if LEN(L_i) > LEN(R_j) then
                    L' ← L' + L_i
                    i ← i + 1
                else
                    R' ← R' + R_j
                    j ← j + 1
                end if
            end while
            L', R' ← L' + L', R' + R'
        end if
    end while
    return L', R'
```

MATCHED CASES:

$$\text{CASE } 1_{(i,j)} : \mathcal{L}_{i(\not\perp)} == \mathcal{R}_{j(\not\perp)} \tag{1}$$

$$\text{CASE } 2_{(i,j)} : (\mathcal{L}_{i(\not\perp)} \simeq \mathcal{R}_{j(\not\perp)}) \wedge (\mathcal{L}_{i+1(\not\perp)} == \mathcal{R}_{j+1(\not\perp)} \vee \mathcal{L}_{i+1(\not\perp)} \simeq \mathcal{R}_{j+1(\not\perp)})) \tag{2}$$

Soundness The algorithm is sound when each premise is true and can return a true answer, resulting in a tautology. By the Soundness Theorem, the Algorithm A is sound through the following three cases. Let $\mathcal{L}, \mathcal{R}, L', R' \in A$.

Case

Algorithm is sound if $L_{\not\perp} \vdash L$ *implies* $L_{\not\perp} \models_{taut} L$ *and* $R_{\not\perp} \vdash R$ *implies* $R_{\not\perp} \models_{taut} R$.

Case

Algorithm is sound if $(L_{\not\perp} \& L_{\sqcup}) \vdash L$ *implies* $(L_{\not\perp} \& L_{\sqcup}) \models_{taut} L$ *and* $(R_{\not\perp} \& R_{\sqcup}) \vdash R$ *implies* $(R_{\not\perp} \& R_{\sqcup}) \models_{taut} R$.

Case

Algorithm is sound if $L_{\not\perp} \vdash L'$ *implies* $L_{\not\perp} \models_{taut} L'$ *and* $R_{\not\perp} \vdash R'$ *implies* $R_{\not\perp} \models_{taut} R'$.

Correctness To prove the correctness of $L' == R'$, let $L' = L_i ... L_n$ and $R' = R_j ... R_m$ for some $i, j, n, m \in \mathbb{Z}^{\geq 0}$.

Case

$L_{i+1} == R_{j+1}$. *For any* i *and* j, *we know that* $i + 1, j + 1 \geq 0$ *such that* $i, j \in \mathbb{Z}^{\geq 0}$. *Hence, for* L_{i+1} *and* R_{j+1} *to have the same value,* i *and* j *will remain in the domain. This will allow* L_{i+1} *and* R_{j+1} *to accumulate together.*

Case

$L_{n+1} == R_{m+1}$. *We know for any* n *and* m, $n + 1$ *and* $m + 1$ *are positives. Since* $n, m \in \mathbb{Z}^{\geq 0}$, L_{n+1} *and* R_{m+1} *are able to accumulate together.*

Therefore, the statement $L' == R'$ is true.

Experiments and Results

Case study on English corpora For our case studies, we conduct preprocessing on five raw input files sourced from English Universal Dependencies (Nivre et al., 2016, 2020). These files include:

1. (1) Universal Dependencies syntax annotations from the `GUM` corpus.
2. (2) A multilingual parallel treebank known as `ParTUT`, developed at the University of Turin.
3. (3) A gold standard Universal Dependencies corpus for English, constructed using the source material of the English Web Treebank (`EWT`).
4. (4) The English portion of the parallel Universal Dependencies (`PUD`) treebanks.
5. (5) The English half of the `LinES` Parallel Treebank.
6. (6) A dataset specifically created for pronoun identification (`Pronouns`).

		GUM			ParTUT			EWT			PUD			LinES			Pronouns		
		TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN
sbd	`nltk & jp`	845	137	251	149	2	4	1084	349	993	976	26	24	865	141	170	285	0	0
	`stanza & jp`	1038	38	58	144	6	9	1805	160	272	998	4	2	912	128	123	285	0	0
	`stanza`* & conllu	1038	38	58	144	6	9	1805	160	272	998	4	2	912	128	123	285	0	0
tk	`nltk & jp`	19443	351	462	3345	27	63	24176	1109	918	20733	251	443	17571	56	104	1673	16	32
	`stanza & jp`	19791	118	114	3381	20	27	24724	286	370	21162	18	14	17517	380	158	1649	28	56
	`stanza`* & conllu	19791	118	114	3381	20	27	24724	286	370	21162	18	14	17517	380	158	1649	28	56

Table 1: Numbers of TP (true positive), FP (false positive) and FN (false negative) using `nltk` and `stanza` for sentence boundaries and tokens. The `stanza*` line provides result numbers by the CoNLL-U evaluation script.

- The lack of consensus on tokenization across different corpora, including Universal Dependencies, contributes to the mismatch issue. Notably, EWT tokenizes *can't* as *ca* and *n't*, while ParTUT tokenizes it as *can* and *not*.
- The same issue can arise when converting "starting quotes" (``) and "ending quotes" ('') in the corpus into straight quotes (") in `nltk`, resulting in discrepancies.
- Since the number of contractions and symbols to convert, such as quotes, in a language is limited, we have created an exception list for our system to capture such cases in English.

Case study on Multilingual corpora

		DE			ES			FR			ID			JA			KO			PT		
		TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN
sbd	`stanza & jp`	787	110	190	373	35	53	393	23	23	515	35	42	541	5	2	629	143	360	1161	71	39
	`stanza`* & conllu	787	110	190	373	35	53	393	23	23	515	35	42	541	5	2	629	143	360	1161	71	39
tk	`stanza & jp`	16172	61	52	11705	33	28	9714	17	22	11523	9	18	12750	361	284	11332	184	345	29311	52	50
	`stanza`* & conllu	16172	61	52	11705	33	28	9710	19	23	11523	9	18	12750	361	284	11332	184	345	29311	52	50

Table 2: Numbers of TP , FP and FN using `stanza` for sentence boundaries and tokens for multilingual case studies using `UD_*-GSD` where `*` is German, Spanish, French, Indonesian, Japanese, Korean, and Portuguese.

- In French GSD, tokenization often combines several units into a single token. For example, the expression *de 1 000 mètres* ('of 1,000 meters') is treated as three tokens instead of four, as spaces are used to separate the words. Notably, this discrepancy is not a limitation of the proposed algorithm but rather stems from differences in tokenization conventions between plain text and the *rich* CoNLL-U format. Even when the input text is *de 1 000 mètres*, with *1* and *000* separated, it is tokenized as separated in plain text.
- Such examples occur across various treebanks. For example, in `UD_Kurmanji-MG`, phrases like *dagir kiriye* ('occupied') are tokenized as a single unit.

Discussion and limitation

The effectiveness of the proposed word alignment approach would remain unaffected even in the presence of significant morphological mismatches.

gold	${}^{0}B$	${}^{1.0}H \sim\sim\sim {}^{1.1}CL$	${}^{2}FL$	${}^{3}HM$	${}^{4.0}H \sim\sim\sim {}^{4.1}NEIM$	(Tsarfaty et al., 2012)
	'in'	'the' 'shadow'	'of'	'them'	'the' 'pleasant'	
sys	${}^{0}B$	${}^{1}CL$	${}^{2}FL$	${}^{3}HM$	${}^{4}HNEIM$	
	'in'	'shadow'	'of'	'them'	'made-pleasant'	

Pairs of $\{{}^{1.0}H\ {}^{1.1}CL, {}^{1}CL\}$ ('the shadow') and $\{{}^{4.0}H\ {}^{4.1}NEIM, {}^{4}HNEIM\}$ ('the pleasant') are word-aligned using the proposed algorithm, and we can obtain 4/5 and 4/7 for precision and recall using the proposed method.

Conclusion

All codes and results from the case studies can be accessed at

`https://github.com/jungyeul/jp-preprocessing/`.

References

Evang, K., Basile, V., Chrupała, G., and Bos, J. (2013). Elephant: Sequence Labeling for Word and Sentence Segmentation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1426, Seattle, Washington, USA. Association for Computational Linguistics.

Gillick, D. (2009). Sentence Boundary Detection and the Problem with the U.S. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 241–244, Boulder, Colorado. Association for Computational Linguistics.

Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., and McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. In Bontcheva, K. and Zhu, J., editors, *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C. D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. (2016). Universal Dependencies v1: A Multilingual Treebank Collection. In von Ahn, L., editor, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, page 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

Nivre, J., de Marneffe, M.-C., Ginter, F., Hajič, J., Manning, C. D., Pyysalo, S., Schuster, S., Tyers, F., and Zeman, D. (2020). Universal Dependencies v2: An Evergrowing Multilingual Treebank Collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.

Tsarfaty, R., Nivre, J., and Andersson, E. (2012). Joint Evaluation of Morphological Segmentation and Syntactic Parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 6–10, Jeju Island, Korea. Association for Computational Linguistics.