MASSAlign: Alignment and Annotation of Comparable Documents

Gustavo H. Paetzold and **Fernando Alva-Manchego** and **Lucia Specia**
Department of Computer Science
University of Sheffield, UK
{g.h.paetzold,f.alva,l.specia}@sheffield.ac.uk

Abstract

We introduce MASSAlign: a Python library for the alignment and annotation of monolingual comparable documents. MASSAlign offers easy-to-use access to state of the art algorithms for paragraph and sentence-level alignment, as well as novel algorithms for word-level annotation of transformation operations between aligned sentences. In addition, MASSAlign provides a visualization module to display and analyze the alignments and annotations performed.

1 Introduction

The ever-growing amount of information produced and distributed electronically has introduced a new challenge: adapting such information for different audiences. One may want, for example, to make their content available for speakers of as many languages as possible, or to make it more accessible for those with reading difficulties, such as those suffering from dyslexia and aphasia, or who are not native speakers of the language.

With that in mind, certain government institutions and content providers produce multiple versions of documents. The result are thousands of pairs of comparable articles, stories and other types of content that render the same information in different ways. Some examples are the European Parliament proceedings[1], which contains translated versions of speeches and other official communications, the Simple English Wikipedia[2], which offers simplified versions of Wikipedia articles; and the Newsela corpus (Xu et al., 2015), which provides versions of news articles for readers with various education levels.

[1] www.europarl.europa.eu
[2] http://simple.wikipedia.org

This data is very useful in the context of Natural Language Processing (NLP): it can be used in the training of automatic translators, simplifiers and summarizers that automate the process of adapting content. In order to do so, machine learning algorithms benefit from texts aligned at lower levels, such as paragraph, sentence, or even word levels. These alignments are however challenging to obtain since documents often do not even have the same number of sentences, i.e. they are *comparable* but not *parallel*. For monolingual texts, which are the focus of this paper, previous work has proposed different ways for obtaining sentence alignments: Xu et al. (2015) extract alignments based on a similarity metric, while Barzilay and Elhadad (2003) employ a more complex data-driven model, and Paetzold and Specia (2016) employ a vicinity-driven search method. However, we were not able to find any available and easy-to-use tool that allows one to align comparable documents at different levels of granularity. To solve that problem, we introduce MASSAlign: a user friendly tool that allows one to align monolingual comparable documents at both paragraph and sentence level, annotate words in aligned sentences with transformation labels, and also visualize the output produced.

2 System Overview

MASSAlign is a Python 2 library. It offers four main functionalities, which we describe in what follows: alignment at paragraph and sentence levels, word-level annotation of transformation operations, and output visualization.

2.1 Paragraph and Sentence Alignment

The `alignment` module of MASSAlign finds equivalent paragraphs and sentences in comparable documents. This module receives as input a pair of documents split at paragraph level and produces as output a series of paragraph alignments,

as well as sentence alignments within the aligned paragraphs. These alignments can be used in the creation of paragraph and sentence-level parallel corpora, which in turn can be employed in the training of models using machine learning.

The alignment method used by MASSAlign is that of Paetzold and Specia (2016), which employs a vicinity-driven approach. The algorithm first creates a similarity matrix between the paragraphs/sentences of aligned documents/paragraphs, using a standard bag-of-words TF-IDF model. It then finds a starting point to begin the search for an alignment path. The starting point is the coordinate in the matrix that is closest to [0,0] and holds a similarity score larger than α, which represents the minimum acceptable similarity for an alignment. They use $\alpha = 0.2$ for their experiments. From the starting point, it iteratively searches for good alignments in a hierarchy of vicinities. In each iteration, the alignment first checks if there is at least one acceptable alignment in the first vicinity. If so, it adds the coordinate with the highest similarity within the vicinity to the path. If not, it does the same to a second vicinity, then a third, and so on. The algorithm ends when it either (i) reaches one of the edges of the matrix, or (ii) fails to find an acceptable alignment. In their experiments, they use three vicinities. Given a coordinate $[i, j]$, they define its first vicinity as $V_1 = \{[i, j+1], [i+1, j], [i+1, j+1]\}$, its second vicinity as $V_2 = \{[i+1, j+2], [i+2, j+1]\}$, and its third vicinity V_3 as all remaining $[x, y]$ where $x > i$ and $y > j$.

We choose this alignment method for various reasons. First, it is one of the few that employs a hierarchical alignment approach, i.e. it exploits information from higher-level alignments to support and improve the quality of lower-level alignments. Moreover, the method can be used in documents that are not organized as a set of paragraphs: one can simply take each comparable document as a large paragraph and then apply the sentence-level alignment algorithm. The method is also entirely unsupervised and one can easily customize the alignment process by changing the similarity metric, the threshold α, or the sets of vicinities considered. Finally, this method has already been shown effective in Paetzold and Specia (2017), where it is used in the extraction of complex-to-simple word pairs from comparable documents to build lexical simplification models.

2.2 Word-Level Annotation

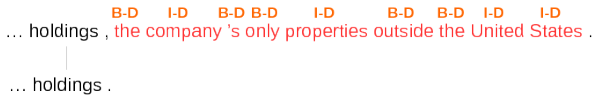
Once paragraphs and sentences have been aligned, one can analyze the differences between the two versions. For example, one can see that a sentence from an original news article was simplified into two others. Furthermore, MASSAlign allows one to obtain insights with respect to which transformation operations were performed at phrase or word-level. Some examples of operations include deletions, where words and/or phrases are discarded; and lexical simplifications, where words and/or phrases are replaced with more familiar alternatives. MASSAlign's `annotation` module provides novel algorithms that automatically identify deletions, substitutions, re-orderings, and additions of words and phrases.

The `annotation` module requires a pair of aligned sentences, their constituency parse trees, and the word alignments between them. To obtain word alignments, many consolidated tools can be employed, such as Giza++ (Och and Ney, 2003), fast_align (Dyer et al., 2013), and the monolingual word aligner (Sultan et al., 2014). Our annotation algorithms only require that the word alignments be in 1-index *Pharaoh* format, which can be obtained from any of the previously mentioned tools.

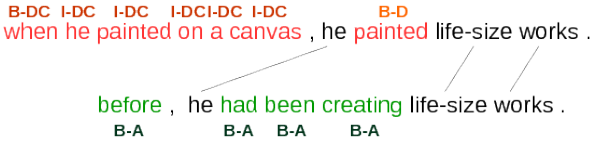
Our module first annotates word-level substitutions, deletions and additions: if two words are aligned and are not an exact match, the word in the original sentence receives a REPLACE tag; if a word in the original sentence is not aligned, it is annotated as a DELETE; and if a word in the modified sentence is not aligned, it is annotated as an ADD. There may be some cases of substitutions where two synonymous are not aligned. In order to improve the REPLACE labeling, we employ a simple heuristic: for every word in the original sentence labeled as DELETE, we check if there is a word in the modified sentence that (1) is labeled as ADD, (2) has the same position in the sentence, and (3) has the same part-of-speech tag. If these criteria are met, then the word label is changed to REPLACE. We also consider REWRITE as a special case of REPLACE or ADD where the words involved are isolated (i.e. no other word with the same label is next to it) and belong to a list of non-content words that we collected after a manual inspection of sample sentences.

We then proceed to labeling re-orderings (MOVE) by determining if the relative index of a word (considering preceding or following DELETEs and ADDs) in the original sentence changes in the modified one. Words that are kept, replaced or rewritten may be subject to re-orderings, such that a token may have more than one label (e.g. REPLACE and MOVE). For that, we extend the set of operations by the compound operations REPLACE+MOVE (RM) and REWRITE+MOVE (RWM).

In order to capture operations that span across syntactic units, such as phrases (chunks) or clauses, we group continuous operation labels for entire syntactic units using IOB notation. The constituent parse trees of the aligned sentences are used for this purpose. If the majority[3] of words within a syntactic unit in the sentence have the same label, the whole unit receives an operation label (for example, DELETE CLAUSE (DC)). We use this algorithm to label clauses and chunks[4], but in the latter case we do not use a particular unit label, and only rely on the IOB notation for the operation labels. Figure 1a presents an example of a DELETE labeling in chunks, while Figure 1b shows the unit label DELETE CLAUSE.



(a) Annotation of operations spanning chunks.



(b) Annotation of operations spanning a clause.

Figure 1: Examples of annotated sentence pairs where an operation label spans across a syntactic unit (chunk or clause).

For evaluation, we compared the algorithms' labels to manual annotations for 100 automatically aligned sentences of the Newsela corpus (Xu et al., 2015)[5]. This corpus consists of news articles and their simplifications, produced manually by professional editors. We achieved a micro-averaged F_1 score of 0.61. For 30 of those sentences, we calculated the pairwise inter-annotator agreement for 4 annotators, with average kappa = 0.57. The annotation algorithms are mainly effective at identifying additions, deletions and substitutions.

2.3 Visualization

The alignments and annotations produced by MASSAlign can be used not only for the creation of parallel corpora, but also for analysis purposes. One can, for example, inspect the sentence alignments between original and simplified documents to find which types of syntactic and semantic transformations with respect to content were made throughout the simplification process. To that purpose, MASSAlign provides a minimalistic graphical interface through its `visualization` module that exhibits paragraph and sentence alignments, as well as word-level annotations. Figures 2 and 3 illustrate these functionalities.

3 Demo Outline

Our demo will be combined with a poster which will show the functionalities of MASSAlign by illustrating how the tool can be used to create parallel corpora for text simplification. Participants will be able to test MASSAlign by producing and displaying alignments and annotations for different kinds of comparable documents on the fly.

4 Discussion and Future Work

We introduced MASSAlign: a Python 2 library that provides tools for the alignment, annotation and analysis of comparable monolingual documents. By using effective methods, MASSAlign is capable of aligning comparable documents at both paragraph and sentence level, annotating aligned sentences at word-level with fine-grained transformation labels, and displaying the alignments and annotations produced in an intuitive fashion.

Through these tools, MASSAlign can create parallel corpora from comparable documents and allow one to analyse the differences between them. MASSAlign was developed following simple software engineering principles such that it can be easily extended with new alignment, annotation and visualisation methods.

In the future, we aim to add to MASSAlign other supervised and unsupervised sentence-level alignment methods, such as the ones of Xu et al.

[3]We consider "majority" as at least 75%, to counteract the effect of incorrect labels caused by word misalignments.

[4]Our definition of "chunk" follows that of the CoNLL 2000 Shared Task: `http://www.cnts.ua.ac.be/conll2000/chunking`.

[5]The Newsela Article Corpus was downloaded from `https://newsela.com/data`, version 2016-01-29.

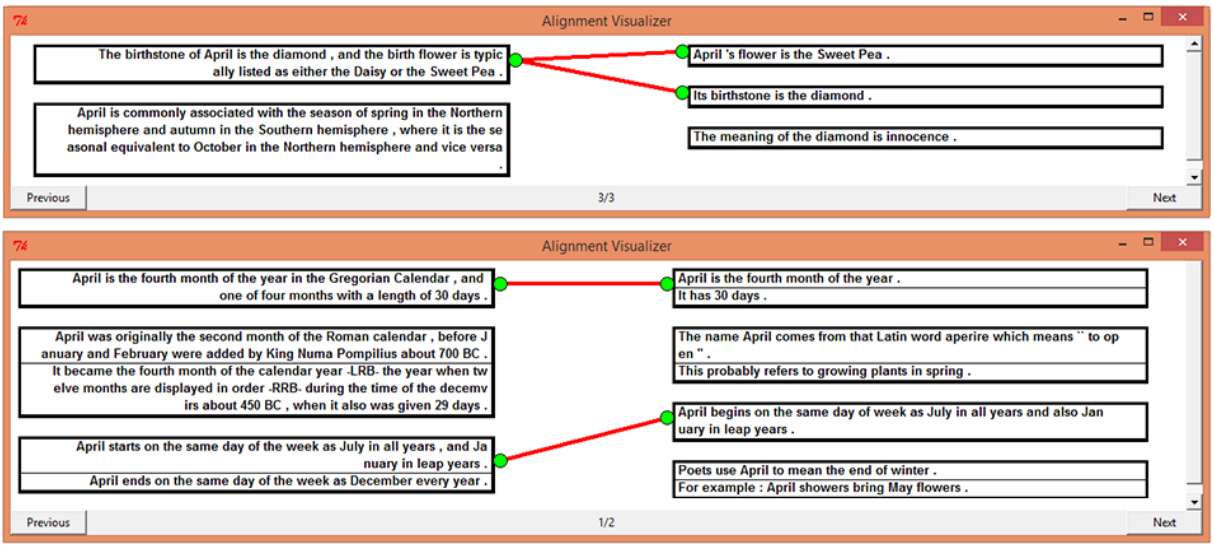


Figure 2: MASSAlign's visualisation interface for alignments.

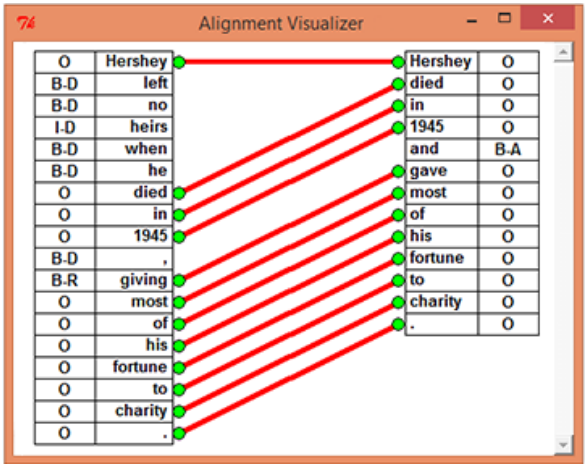


Figure 3: MASSAlign's visualisation interface for annotations.

(2015), Kajiwara and Komachi (2016), Bott and Saggion (2011), and Barzilay and Elhadad (2003), as well as built-in word alignment methods, such as the ones in (Dyer et al., 2013) and (Sultan et al., 2014). By doing so, the tool will become more self-contained and more flexible.

MASSAlign is available for download at `https://github.com/ghpaetzold/massalign` under a BSD license.

Acknowledgements

This work was partly supported by the EC project SIMPATICO (H2020-EURO-6-2015, grant number 692819).

References

Regina Barzilay and Noemie Elhadad. 2003. Sentence alignment for monolingual comparable corpora. In *Proceedings of EMNLP*, pages 25–32.

Stefan Bott and Horacio Saggion. 2011. An unsupervised alignment algorithm for text simplification corpus construction. In *Proceedings of the 2011 MTTG*, pages 20–26.

Chris Dyer, Victor Chahuneau, and Noah Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of NAACL*, pages 644–648.

Tomoyuki Kajiwara and Mamoru Komachi. 2016. Building a monolingual parallel corpus for text simplification using sentence similarity based on alignment between word embeddings. In *Proceedings of COLING*, pages 1147–1158.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Gustavo H. Paetzold and Lucia Specia. 2017. Lexical simplification with neural ranking. In *Proceedings of EACL*, pages 34–40.

Gustavo Henrique Paetzold and Lucia Specia. 2016. Vicinity-driven paragraph and sentence alignment for comparable corpora. *arXiv preprint arXiv:1612.04113*.

Md Sultan, Steven Bethard, and Tamara Sumner. 2014. Back to basics for monolingual alignment: Exploiting word similarity and contextual evidence. *TACL*, 2:219–230.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *TACL*, 3:283–297.