Computational Linguistics and Chinese Language Processing
Vol. 27, No. 2, December 2022, pp. 1-30

Aligning Sentences in a Paragraph-Paraphrased Corpus with New Embedding-based Similarity Measures

Aleksandra Smolka*, Hsin-Min Wang+,

Jason S. Chang#, and Keh-Yih Su+

Abstract

To better understand and utilize lexical and syntactic mapping between various language expressions, it is often first necessary to perform sentence alignment on the provided data. Up until now, the character trigram overlapping ratio was considered to be the best similarity measure on the text simplification corpus. In this paper, we aim to show that a newer embedding-based similarity metric will be preferable to the traditional SOTA metric on the paragraph-paraphrased corpus. We report a series of experiments designed to compare different alignment search strategies as well as various embedding- and non-embedding-based sentence similarity metrics in the paraphrased sentence alignment task. Additionally, we explore the problem of aligning and extracting sentences with imposed restrictions, such as controlling sentence complexity. For evaluation, we use paragraph pairs sampled from the Webis-CPC-11 corpus containing paraphrased paragraphs. Our results indicate that modern embedding-based metrics such as those utilizing SentenceBERT or BERTScore significantly outperform the character trigram overlapping ratio in the sentence alignment task in the paragraph-paraphrased corpus.

Keywords: Sentence Alignment, Sentence Similarity, Sentence Embedding, Paragraph-paraphrased Corpus

* Social Networks and Human Centered Computing, Taiwan International Graduate Program
Institute of Information Science, Academia Sinica, Taipei, Taiwan
E-mail: aleksandra.smolka@hotmail.com
+ Institute of Information Science, Academia Sinica, Taipei, Taiwan
E-mail: {whm, kysu}@iis.sinica.edu.tw
Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan
E-mail: jason@nlplab.cc

1. Introduction

Monolingual text matching is necessary for many downstream applications, such as Paraphrase Identification and Extraction (Qiu *et al.*, 2006), Question Answering (Weiss *et al.*, 2021), Natural Language Inference (MacCartney & Manning, 2008), and Text Generation (Barzilay & McKeown, 2005). Take the QA task as an example, identifying the text fragments that match the given question within the associated passage is often required for locating the desired answer.

However, modern neural network (NN) approaches to text matching often suffer from certain limitations when two sequences contain considerably different lexicons or diverse grammatical structures (McCoy *et al.*, 2019). For example, when the verb "*decide*" in the sentence "*They **decided** to go*" is nominalized to the noun "*decision*" in its paraphrase "*They **made a decision** to go*", the popular word embedding similarity approach might fail as the embedding-vectors of "*decide*" and "*decision*" are quite different[1]. Another example is a pair of sentences "*A cat is chasing a dog*." and "*A dog is chasing a cat*.", which contain the same set of lexicons and syntactic structure but with opposite meanings.

Furthermore, the NN approaches frequently fail when the matching involves multi-word expressions, or when expressions require compositionality handling (Blevins *et al.*, 2018; Hupkes *et al.*, 2020; Zhou *et al.*, 2020). For example, it is difficult to match expressions "*put off*" and "*procrastinate*" using basic word embeddings, as the real meaning of the idiom "*put off*" is not the sum of the meanings of its tokens.

We found that the limitations of NN models in text matching could be greatly alleviated by utilizing lexico-syntactic paraphrasing patterns such as $[_{VP}[_{VBN}[see]_{NP}[X1]]] \rightarrow [_{S}[_{NP}[X1]_{VP}[_{VBD}[be]_{VP}[observe]]]$, which denotes the conversion from active to passive voice for the phrase pair "*see the lion*" and "*the lion is observed*". Since some key lexicons are involved in the pattern, it would be difficult to exhaustively list such patterns by a human. It is preferable to automatically extract them from a large paraphrase corpus.

To collect such lexico-syntactic patterns, a high-quality paraphrased sentence pair dataset is essential. Unfortunately, current *sentence-aligned* paraphrase datasets (e.g., MRPC (Dolan & Brockett, 2005), PPDB (Ganitkevitch *et al.*, 2013), and QQP (Aghaebrahimian, 2017)) are too trivial for this task, as they mainly contain lexical paraphrases that could be easily handled by a NN. On the other hand, some *paragraph-aligned* paraphrase corpora, containing different human translations from the same source text, fit our needs well. To utilize those paragraph-

[1] The nearest semantic associates of the verb *decide* based on the cosine similarity between the word2vec vectors (trained on English Wikipedia) are those verbs such as: *choose* (0.64), *opt* (0.62), *persuade* (0.61), *want* (0.58), *refuse* (0.57), *insist* (0.56). However, the noun *decision* only has a similarity score 0.512, which means that its similarity to the verb *decide* is even less than that between *decide* and its quasi-antonymous *refuse*.

aligned paraphrase corpora, monolingual sentence alignment is the first step in retrieving the desired patterns.

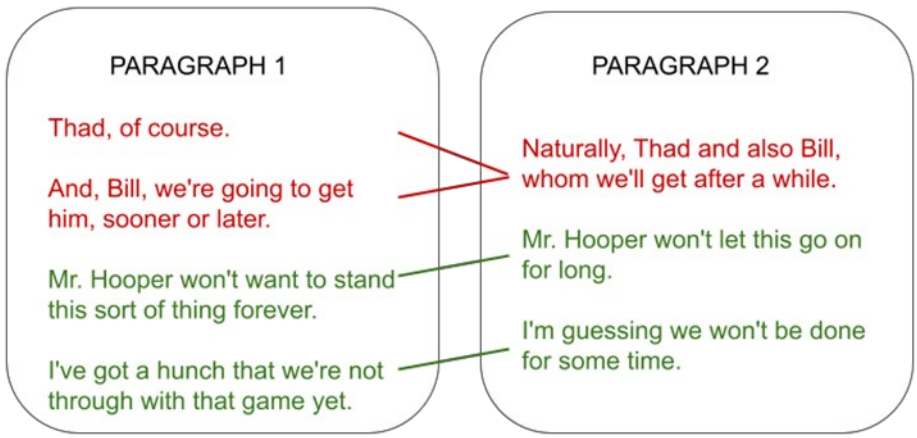

Figure 1. Sentence alignment for extracting paraphrased sentence pairs. Sentence pairs in green are those we want to extract; sentences in red are in multi-to-one relation and do not constitute sentential paraphrases. Figure adopted from Smolka et al. (2022).

Figure 1 shows how a correct sentence alignment could help extract paraphrased sentence pairs from longer paraphrased texts. Unless we correctly identify which sentences are in 1-to-1 relationships (green in the figure), we cannot correctly identify the desired paraphrased pattern.

Monolingual sentence alignment approaches could be classified into two categories: *model-based* approaches (e.g., Jiang *et al.*, 2020), which adopt specific models to encode the input sentences and perform alignment, and *model-agnostic* approaches (Štajner *et al.*, 2018), which can be directly applied to the selected dataset, without the necessity of training a neural model in advance. In our work, we focus on model-agnostic approaches, as they do not require additional labeled data to train the model.

The downside of previous model-agnostic approaches (Štajner *et al.*, 2017; 2018) is that they only test the early word2vec word embeddings, and do not explore those more advanced NN approaches such as Sentence-BERT (Reimers & Gurevych, 2019) and BERTScore (Zhang *et al.*, 2020). Also, they are mainly evaluated on Text Simplification (TS) datasets, which are different from our paraphrasing datasets.

In the TS dataset, the original and the simplified text often share a considerable number of keywords, which remain unchanged and are rarely substituted with synonyms. However, this property does not hold in our paraphrasing corpus, as its paraphrasing expressions usually possess diverse syntactic structures with many different lexical items.

Therefore, we suspect that the character trigram overlapping ratio, reported as the best for monolingual sentence alignment in previous works (Štajner *et al.*, 2017; 2018), would not perform best on our data. Since our paraphrasing corpus contains considerably different lexicons and word order, the string-based method such as character ngram similarity would lose its edge. Previously reported text similarity measures thus should be re-evaluated for our task, and more advanced NN approaches should be explored.

In this work, we not only compare various previously reported text similarity measures on a paraphrased paragraph corpus but also additionally test some new measures based on the most recent NN sentence embedding methods. We utilize those above measures with two sentence alignment approaches: simple greedy match (e.g., Štajner *et al.* 2018) and sequence match (Gale & Church, 1993; Barzilay & McKeown, 2001). We conduct the evaluation on a manually annotated sentence-aligned dataset with 400 paraphrased paragraph pairs randomly sampled from the multiple translation corpus Webis-CPC-11 (Burrows *et al.*, 2013).

Our contributions include:

(1) To the best of our knowledge, we present the first study on aligning sentences on a paragraph-paraphrased corpus;

(2) We show that character trigram similarity is not the best measure for aligning paraphrasing corpora. Instead, BERT-based embedding methods achieve significantly better results even without fine-tuning on the target dataset;

(3) We test several NN-related sentence similarity measures (other than word2vec) that have not been evaluated before for model-agnostic monolingual sentence alignment;

(4) We confirm and expand the observation of Choi *et al*., (2021), showing that [CLS] token representation is not necessarily superior to averaging individual word vectors for sentence representation while aligning paraphrased text under BERT.

(5) We compare the sentence alignment methods when an additional sentence length limitation is imposed on the data.

This publication is an extension of our previous conference paper on the same topic (Smolka *et al.*, 2022). In comparison to our conference publication, we have added a new data collection method for composing a dataset with sentence length limitation and introduced a new series of experiments performed on this new data (Section 3.4.2). We also extend the previous comparison of different methods of obtaining sentence representations using the BERT model (Section 3.4.3), and add a new discussion section to report our observations (Section 5). Finally, we extend some of the previously existing sections by additionally illustrating our search mechanisms (Section 2.1, Figure 2), showing an example of a non-paraphrased paragraph pair (Section 3.5, Figure 5), and a new error example in the error analysis (Section 4, Table 12).

2. Sentence Alignment Procedure

The proposed sentence alignment procedure is based on two basic elements, which we combine to test different experimental configurations. Those elements include: (1) search mechanism, which specifies the method used to search the sentence pairs that possess similar meaning; (2) similarity measure, which defines the method of calculating the similarity value among two given sentences (to be used during the search procedure).

2.1 Search Mechanisms

We implement two search mechanisms for aligning sentences among two paraphrased paragraphs: (1) *Directional Best Match*, which aligns each sentence in the paragraph separately (it also has two variations: *uni-directional*, which matches sentences from the first paragraph to the second one only, and *bi-directional*, which matches sentences in both directions), and (2) *Sequence Match*, which looks for the best alignment scheme for a paragraph as a whole. Figure 2 schematically illustrates the difference in how the sentence pairs are formed in the different search mechanism approaches, which we describe in the next two subsections.

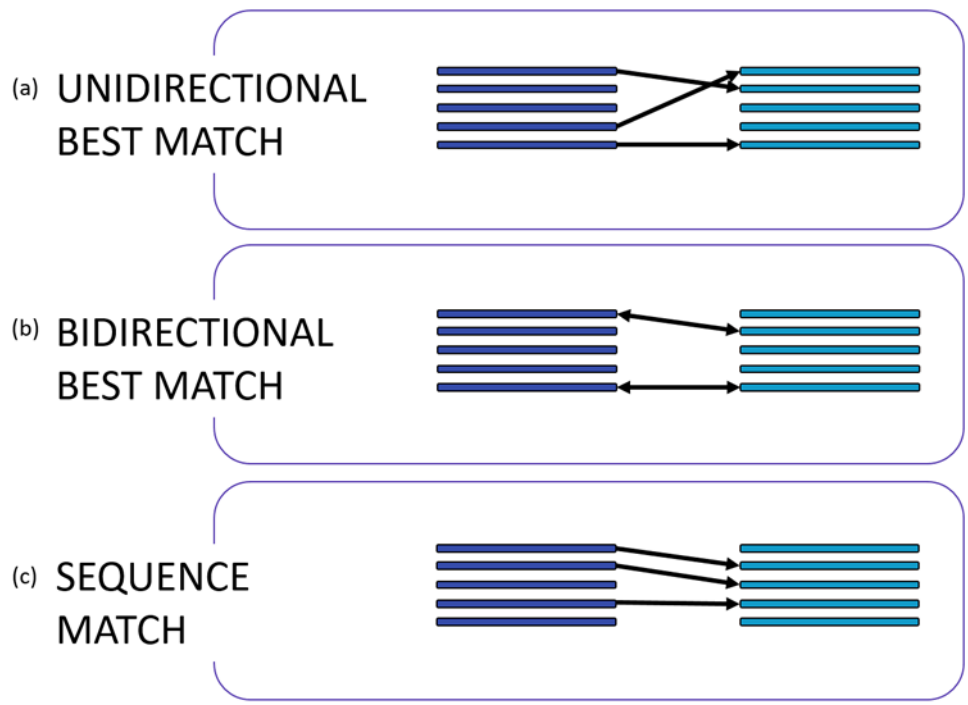

Figure 2. Schematic comparison of the different search mechanisms, illustrating the direction in which sentences are paired. (a) Uni-directional Best Match; (b) Bi-directional Best Match; (c) Sequence Match.

2.1.1 Directional Best Match

Directional Best Match is a simple greedy approach that relies on local judgments to create the alignments. This approach assumes that information such as adjacency and dependency information within sentences is negligible during matching. In our implementation, we follow the adopted SOTA approach (Štajner *et al.*, 2018). However, Štajner *et al.* (2018) only experimented with a uni-directional approach, which maps the original passages to the corresponding simplified passages. We believe that the bi-directional approach would be better applicable to our data since it is symmetric, unlike the dataset used by Štajner *et al.* (2018). Therefore, we additionally extend the Best Match method to a bi-directional approach.

Regardless of the above variation, we first calculate the associated similarity for each sentence pair that can be formed between the two given input paragraphs. Then, for each sentence in one paragraph, we select the sentence in another paragraph that has the highest similarity measure obtained above. For the uni-directional version, we directly take those pairs as the final alignments.

The bi-directional version follows the same steps, but we additionally repeat them in the opposite direction, i.e., matching sentences from the second paragraph to the first one. The final aligned pairs are obtained by taking the intersection of the two sets of aligned sentence pairs.

2.1.2 Sequence Match

Our sequence match adopts the dynamic programming searching algorithm to look for the best alignment path (among the two given paragraphs). Our implementation follows the common approach described in previous works (Gale & Church, 1993; Barzilay & McKeown, 2001). In this method, each alignment type (e.g., one-to-one and one-to-two) is associated with a different weight indicating the type probability estimated from the development set. The weights are then combined with the above similarity measures to find the best alignment path for the whole paragraph.

2.2 Similarity Measures

The text similarity measures adopted in our experiments fall into two main categories: (a) unit-overlap-based approaches, in which the similarity measure is based on the overlapping ratio of either ngrams or tokens between the sentences; (b) sentence-vector-based approaches, in which a neural model is first used to convert each sentence into its corresponding embedding-vector, and then the cosine similarity between these two sentence embedding-vectors is taken as the sentence similarity.

2.2.1 Unit-Overlap-Based Sentence Similarity

We adopt two different overlapping ratios: (1) *Character ngram*, which is reported as the state-of-art on the text simplification corpus (Štajner, 2018), and (2) *token*, which is commonly used in sentence alignment tasks (e.g., Barzilay & McKeown, 2001).

Character Ngram

We follow Štajner *et al.* (2018) to calculate the ngram similarity based on the *Character Ngram Similarity* model with tf-idf weighting (adapted from McNamee & Mayfield (2004)). We experiment with five different ngram sizes (1 to 5) and use NGRAM to refer to this measure. We add Laplace smoothing to account for those unseen ngrams in the test set. The final similarity is calculated by taking cosine similarity (Štajner *et al.*, 2018).

Token

For calculating token-based sentence similarity, we use the following token overlap formula:

$$similarity_{token} = \frac{|tokens_1 \cap tokens_2|}{|tokens_1| + |tokens_2|} \quad (1)$$

where $tokens_1$ is the set of tokens in the first sentence, $tokens_2$ is the set of tokens in the second sentence, and the function | | specifies the cardinality of the token set. We consider two different normalization mechanisms for comparing two tokens: (1) converting the strings into their associated lemmas before comparison (abbreviated as TOKENstring); (2) also taking synonyms as exactly matched lemmas during comparison (abbreviated as TOKENsyn). Token lemmas for each sentence are retrieved using an automatic tokenizer and lemmatizer (Qi *et al.*, 2020). Synonymic relationships are taken from WordNet (Fellbaum, 1998).

2.2.2 Sentence-Vector- Based Sentence Similarity

This category includes similarity measures that utilize cosine vector similarity in some forms: (1) *word-embedding* based, where we first look up the word embedding-vector for every token in each sentence from a pretrained model, and then combine them into their associated sentence embedding-vector by vector averaging (Putra & Tokunaga, 2017). Afterward, we calculate the similarity between the two obtained sentence embedding vectors. (2) *sentence-embedding* based, where we use a model, such as BERT (Devlin *et al.*, 2019) or Sentence-BERT (Reimers & Gurevych, 2019), to directly embed a sentence into its associated sentence-embedding. We then calculate the similarity between these two sentence embedding vectors. (3) *BERTScore* (Zhang *et al.*, 2020), which uses BERT to directly generate the similarity value between two sentences.

Word-embedding Similarity

For directly retrieving the token-associated embedding vector from a pretrained embedding lookup table, we test both word2vec (Mikolov *et al.*, 2013) and Glove (Pennington *et al.*, 2014)

embeddings. Additionally, we also test contextualized word embeddings retrieved from BERT (Devlin *et al.*, 2019).

Moreover, while it is common to use the [CLS] token yielded by the BERT encoder to represent the whole encoded sentence, recent works note that this might not be the best solution for different downstream tasks (Choi *et al.*, 2021). We therefore additionally test the following approach: generate the sentence embedding via averaging the contextual word embeddings retrieved from the BERT model.

Regardless of the way of selecting word embedding, we combine the associated embedding vectors into the corresponding sentence representation by taking an average over them (Putra & Tokunaga, 2017). The sentence similarity is then calculated as the cosine similarity between the two sentence embedding vectors.

Among various types of word embeddings, only word2vec is tested by Štajner *et al.* (2018). However, it was not reported as the best one in their experiments (the best one is the character trigram in their task).

Sentence-embedding Similarity

Another way to generate the sentence-embedding is to adopt BERT to transform all its associated token-embeddings into it. We test two methods of obtaining sentence representation via BERT. First, we take the [CLS] token from the BERT to represent the whole sentence. Alternatively, we use Sentence-BERT (Reimers & Gurevych, 2019), which is an alternative method of obtaining sentence representation from BERT-type models, suggested as a better alternative for directly adopting [CLS] token embedding. We use Sentence-BERT to separately obtain a single embedding for each sentence in the pair. The sentence similarity is then calculated between two obtained sentence embedding vectors.

BERTScore

Last, we can directly generate the desired similarity value among two sentences by adopting the BERTScore (Zhang *et al.*, 2020) approach, which is originally developed as an automatic evaluation metric for comparing various text generation systems. This approach first uses BERT to obtain the word embeddings of all input tokens. The pairwise similarity is then calculated for each possible token pair. Afterward, for each token from the first input sequence (i.e., the sentence from the "*original*" paragraph), BERTScore finds its matching token in the second sequence (i.e., the sentence from the "*paraphrased*" paragraph) via greedy search. Last, it calculates both precision and recall based on the matching result.

As BERTScore is designed to evaluate the similarity between the ground truth and the generated text, we thought it should be also suitable for measuring the sentence similarity for our task. Typically, BERTScore will report precision, recall, and F1-score at the same time. We take each of these values to represent a specific sentence pair similarity measure; and we refer

to them as BERTprec, BERTrec, and BERTf1, respectively.

2.3 Similarity Score Thresholding

Regardless of the selected combination of search mechanism and similarity measure, we additionally impose a similarity score thresholding on the aligned sentences. In the final stage of the alignment procedure, we filter out sentence pairs that have similarity values below the experimentally selected threshold. This helps us further improve the overall test-set results and allows for a precision-recall trade-off if desired.

3. Experiments

Figure 3 shows the operation flow adopted in the experiments. We first take a pair of paraphrased paragraphs as input, clean the text in each paragraph, and split it into individual sentences. Then, we use the sentence alignment module with the selected search mechanism and similarity measure to generate the desired sentence alignments. Those one-to-one sentence alignments are then extracted and output as the answer.

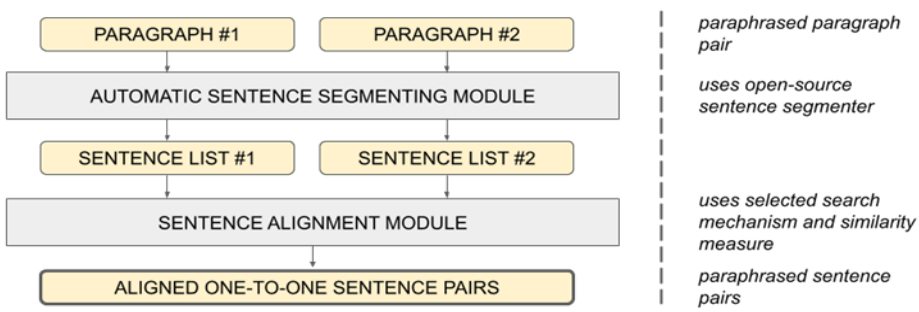

Figure 3. Operation flow for obtaining one-to-one sentence alignment within paraphrased paragraph pairs. Figure adopted from Smolka et al. (2022).

The following subsections give details of the experiment setting and results.

3.1 Dataset

We randomly sampled 400 paragraph pairs from the Webis-CPC-11 corpus (out of which 7 were found to be incorrectly marked as paraphrases, and removed from the evaluation data). The non-paraphrased pairs are excluded from the development and test data. However, we reserve them for additional experiments where we test methods for automatically detecting such undesired input from our data.

To evaluate the performance, we manually annotate the 400 paragraph pairs randomly sampled from the Webis-CPC-11 corpus. The annotation process consists of several stages: (1) Paragraph pre-processing, which is performed automatically and serves to clean the data and

split each paragraph into its associated sentences; (2) Sentence alignment (marking both one-to-one and one-to-many alignment configurations), in which we manually match the sentences that have similar meanings.

After the paragraph pre-processing stage, the annotator receives two sets of sentences for each paragraph pair and is requested to align sentences between them (including both one-to-one and one-to-many mappings). The result of the manual annotation is a dataset in which each paraphrased paragraph pair is associated with the aligned sentence pairs between them. If the sample contains non-paraphrased paragraphs, the annotator is asked to simply mark them without adding alignment annotation.

As all tested similarity measures are model-agnostic, we do not require a training set. Therefore, we split all the aligned paragraph pairs (i.e., excluding those non-paraphrased pairs) into the development set and the test set with a 1:7 ratio. As a result, we end up with 48 paragraph pairs in the development set and 345 paragraph pairs in the test set. We use the development set for selecting hyper-parameters such as similarity cutting threshold and alignment type probabilities for the Gale-Church algorithm (Gale & Church, 1993).

Table 1. Dataset Statistics (without non-paraphrase cases). #Min-#Max specifies the range in paragraph range row. Also, 1-1 indicates the one-to-one mapping, 2-1 (1-2) indicates two-to-one and one-to-two mapping, and so on.

		all	dev	test
#input paragraphs		393	48	345
#input non-paraphrased pairs (dataset errors)		7	2	5
avg. paragraph length (#sentences)		2.3	2.4	2.3
avg. sentence length (#tokens)		20.9	19.3	21.1
paragraph range (# sentences)		1-7	1-6	1-7
% of alignment types	all	822 (100%)	87 (100%)	735 (100%)
	1-1 (ground truth)	**633 (77%)**	**67 (77%)**	**566 (77%)**
	2-1 (1-2)	132 (16%)	16 (18%)	118 (16%)
	2-2	8 (1%)	1 (1%)	7 (1%)
	Other (2-3,1-4,etc.)	49 (6%)	4 (4%)	45 (6%)

Table 1 gives the associated dataset statistics. Within them, 566 1-to-1 paraphrased sentence pairs (77% among all aligned passage pairs) exist in the test set. This set of 1-to-1 sentence pairs (i.e., sentential paraphrases) is the desired output in our task, and thus becomes the ground truth for our evaluation.

3.2 Pre-processing

Because the Webis-CPC dataset only contains un-segmented paragraphs, it must be first converted into a collection of sentences. We use an off-the-shelf sentence segmenter (Qi *et al.*, 2020) to split each paragraph into sentences. The output is thus two sets of sentences, one for each of the paragraphs.

3.3 Experimental Setting

For our baseline, we re-implement the SOTA approach proposed by Štajner *et al.* (2018), as there is no easily applicable code released by the authors. Therefore, we follow the descriptions in the original paper to implement the ngram character similarity. Our implementation has not been tested on the data adopted in the work of Štajner *et al.* (2018) because it lacks the annotations that are necessary for automatic evaluation. Furthermore, the original work introducing SOTA character trigram metrics used only human evaluation, which makes a direct comparison of our method with their results impossible.

When it comes to the pretrained models used for conducting the embedding-based similarity calculations, we select the models based on their open-source availability. For example, for getting the BERT word-averaging and [CLS]-token representation, we use the BERT-base model (Devlin *et al.*, 2019). When it comes to Sentence-BERT, three different pretrained models were tested, including *BERT-base* (Devlin *et al.*, 2019; abbreviated as SBERTbert), *ALBERT-mini* (Lan *et al.*, 2020; abbreviated as SBERTalbert), and *MiniLM* (Wang *et al.*, 2020; abbreviated as SBERTmini). The training data for those three SentenceBERT models varied and depended on the original open-source model released.[2] Among them, SBERTbert was trained with various Natural Language Inference data sets; SBERTalbert and SBERTmini were trained on various paraphrasing datasets.[3] Finally, the BERTScore open-source implementation uses ROBERTA-Large (Liu *et al.*, 2019).

3.4 Various Experiments

In our experiments, we test various combinations of the two alignment strategies with different similarity measures. We take precision, recall, and F1-score as the evaluation metrics. Moreover,

[2] https://huggingface.co/sentence-transformers

[3] The list of specific datasets used was not published by the open-source authors.

for each set of results, we apply the McNemar test (Dietterich, 1998) to check whether the performance improvement is statistically significant (with $p \leq 0.05$ as the significance test threshold).

In our experiments, we test similarity measures based on: **(1) Unit-Overlap-Ratio**, including character ngram overlap-ratio with *n* ranging from 1 to 5 (NGRAM), and token overlap-ratio calculated with either token strings (TOKENstring) or token synonyms (TOKENsyn); **(2) Sentence-Vector-Similarity**, including **(a) word-embedding-based** similarity measures calculated with word2vec (W2V), Glove (GLOVE) and BERTbase (BERTword) embeddings; **(b) sentence-embedding-based** similarity measures which consist of: (i) using [CLS] token yielded by BERTbase model (BERTcls), and (ii) Sentence-BERT embeddings with three different pretraining models (SBERTbert, SBERTalbert, and SBERTmini); **(c) BERTScore** with precision (BERTprec), recall (BERTrec), and F1-score (BERTf1).

3.4.1 Sentence Alignment Results on the Full Dataset

Tables 2-4 compare various similarity measures under the Best Match (Uni- and Bi-directional, separately) strategy and the Sequence Match strategy, respectively. For each measure, we only report the results with the best threshold value, which is selected on the development set based on the F1 value. The threshold for each specific similarity measure is different and is noted in the corresponding table. Measures that outperform the character trigram baseline in a significant manner are marked with the asterisk *.

Overall, comparing the best result of each approach, the sequence match approach (with the best F1-score equaling 88.8%) outperforms both best match approaches (the best F1-score of 85.1% is from the bi-directional mode). We conjecture that the sequence match performs the best as it additionally considers the adjacency and dependency information within sentences during matching.

Moreover, the Uni-directional Best Match approach performed the worst (only with 82.5% best F1) as expected. Since our data is symmetric, the matching results would be more reliable if the alignment is considered from both directions.

Furthermore, the best similarity measure varies under different search mechanisms. In the sequence match approach, three BERT-type measures (i.e., SBERTbert (88.8% F1), BERTrec (88.7% F1), and BERTf1 (88.7% F1)) significantly outperform the baseline. The SentenceBERT measure performs best, surpassing the character-trigram baseline method by 1.9% (88.8% vs. 86.9%) because it is trained to encode the overall sentence meaning, not the specific meaning of individual tokens, which fits our task well. Similarly, BERTScore also delivers good results because it is directly trained to measure the similarity between two

sequences.

Table 2. Alignment results by adopting the uni-directional Best Match strategy on the full dataset. TH indicates the adopted threshold value. The asterisk * marks the measures that outperform NGRAM baseline (n=3) with p ≤ 0.05. Table adopted from Smolka et al. (2022).

measure	% on the test set			Best *TH*
	prec	rec	F1	
NGRAM(n=1)*	77.8	82.2	79.9	0.3
NGRAM(n=2)*	77.8	82.2	79.9	0.3
NGRAM(n=3)	79.9	72.5	76.1	0.3
NGRAM(n=4)*	77.8	82.2	79.9	0.3
NGRAM(n=5)*	77.8	82.2	79.9	0.3
TOKENstring*	83.7	73.1	78.1	0.2
TOKENsyn	77.1	71.5	74.2	0.1
W2V	79.7	74.5	77.0	0.8
GLOVE	73.5	81.2	77.1	0.95
BERTword*	78.5	**87.0**	**82.5**	0.75
BERTcls	81.9	67.9	74.3	0.9
SBERTbert	75.2	90.8	82.3	0.6
SBERTalbert	82.9	70.7	76.9	0.35
SBERTmini*	78.4	85.2	81.6	0.6
BERTprec*	86.5	72.9	79.1	0.9
BERTrec*	83.5	74.9	80.4	0.9
BERTf1*	**86.8**	74.9	80.4	0.9

On the other hand, in the bi-directional best match approach, the best result is again obtained by the Sentence-BERT measure (SBERTmini) with the best F1-score 85.1%, significantly outperforming the character ngram similarity measure at 82.7%. Also, both SBERTalbert and BERTf1 measures outperform the baseline with $p<0.06$. We believe that the above reasons given for the sequence match approach also apply here.

Table 3. Alignment results by adopting bi-directional Best Match strategy on full dataset. TH indicates the adopted threshold value. The asterisk * marks the measures that outperform NGRAM baseline (n=3) with p ≤ 0.05. Table adopted from Smolka et al. (2022).

measure	% on the test set			Best *TH*
	prec	rec	F1	
NGRAM(n=1)	80.5	81.8	81.1	0.3
NGRAM(n=2)	80.5	81.8	81.1	0.3
NGRAM(n=3)	78.9	87.0	82.7	0.1
NGRAM(n=4)	80.5	81.8	81.1	0.3
NGRAM(n=5)	80.5	81.8	81.1	0.3
TOKENstring	84.7	73.1	78.5	0.2
TOKENsyn	78.6	81.8	80.2	0.05
W2V	81.1	87.6	84.2	0.6
GLOVE	79.7	78.0	78.8	0.95
BERTword	82.3	86.4	84.3	0.75
BERTcls	**86.2**	66.5	75.1	0.9
SBERTbert	79.1	88.6	83.6	0.6
SBERTalbert	80.6	89.8	84.9	0.25
SBERTmini*	80.7	90.2	**85.1**	0.25
BERTprec	80.9	88.2	84.4	0.85
BERTrec	79.7	88.2	83.7	0.85
BERTf1	79.9	**90.8**	85.0	0.9

Last, in the uni-directional best match approach, several tested measures significantly outperform the baseline (76.1%), including BERTword (82.5%), SBERTbert (82.3%), SBERTmini (81.6%), BERTf1(80.4%), NGRAM with n≠3 (79.9%), BERTrec (79.7%), BERTprec (79.1%), and TOKENstring (78.1%). The measures that perform best in this search mechanism are again mostly those that encode the sentence as a whole, similar to other search mechanisms.

Table 4. Alignment results by adopting Sequence Match strategy on the full dataset. TH indicates the adopted threshold value. The asterisk * marks the measures that outperform NGRAM baseline (n=3) with $p \leq 0.05$. Table adopted from Smolka et al. (2022).

measure	% on the test set			Best *TH*
	prec	rec	F1	
NGRAM(n=1)	89.1	83.4	86.1	0.2
NGRAM(n=2)	89.1	83.4	86.1	0.2
NGRAM(n=3)	89.7	84.2	86.9	0.1
NGRAM(n=4)	89.1	83.4	86.1	0.2
NGRAM(n=5)	89.1	83.4	86.1	0.2
TOKENstring	**92.7**	81.6	86.8	0.15
TOKENsyn	86.2	86.9	86.3	0
W2V	87.6	87.6	87.6	0.45
GLOVE	87.3	85.2	86.2	0.9
BERTword	91.5	82.2	86.6	0.75
BERTcls	92.3	81.4	86.5	0.85
SBERTbert*	89.8	**87.8**	**88.8**	0.6
SBERTalbert	91.1	85.8	88.3	0.25
SBERTmini	87.8	86.8	87.3	0.25
BERTprec	90.0	86.8	88.4	0.85
BERTrec*	89.9	87.6	88.7	0.85
BERTf1*	90.1	87.4	88.7	0.85

3.4.2 Alignment Results on Sentences with Limited Length

The above experiments are conducted without limiting the lengths of those input sentences. However, in our another study, we have found that it is difficult to extract appropriate lexico-syntactic patterns from sentences containing more than two clauses, as selecting the desired candidates will become much more confusing. As a result, the precision rate of extracting high-quality patterns would be lower. To ensure the quality of extracted templates, we thus conducted an additional set of experiments on those input sentences with limited length. Below, we first

describe how to find out a reasonable way to filter out those sentences that would be too complicated/long for our purpose. Afterward, we repeat the above experiments on this new dataset to check if it would significantly change the alignment performance.

3.4.2.1 Finding the Appropriate Criterion to Filter out Long Sentences

To limit the degree of confusion in selecting the desired candidates, we would like to only use sentences with no more than two clauses to extract the desired templates. To automatically filter out those sentence pairs that might contain more than two clauses, we need to first find out a suitable criterion. For simplicity, we opt to use sentence length as the filtering criterion, because this value not only is highly correlated with the number of associated clauses but also could be easily measured.

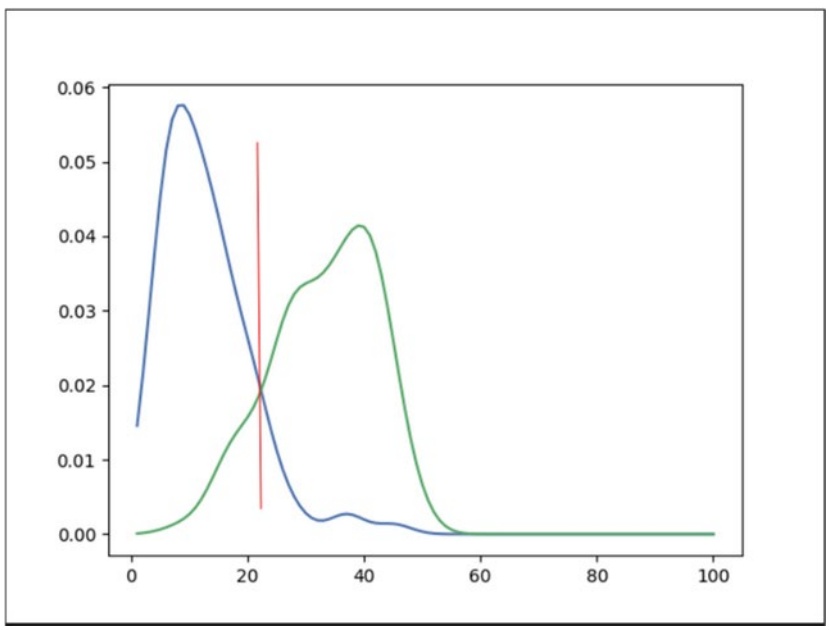

Figure 4. Finding the upper-limit sentence length from smoothed probability distributions (X-axis: sentence length in tokens). The blue curve is for the sentences with maximum two clauses, and the green curve is for the cases with more than two clauses. The red vertical line marks the intersection between the two distributions.

Figure 4 illustrates how we find the upper-limit sentence length. We first manually generate two smoothed probability distributions in Figure 4: The blue curve is for the sentences with two clauses at most (which we consider appropriate for our task), and the green curve is for the cases with more than two clauses (which we consider are too difficult). Those two smoothed probability distributions are constructed from 100 sentences in each group, which are randomly selected from the Webis-CPC-11 dataset and then manually checked to fit this target number. The smoothed probability distributions are calculated using kernel density estimation

(Rosenblatt, 1956). We then find the integer value that is closest to the intersection point between the two distributions (indicated by the red vertical line in Figure 4), which is 22. This value indicates that if the sentence has a length below it, it is more likely to belong to the "appropriate" category. On the contrary, a sentence is more likely to be too difficult for our purpose, if its length is above this value.

Table 5 gives the details of the newly constructed dataset. The main difference from the full dataset used in the previous experiments lies in the golden answers. In the new dataset, the benchmark consists of only 367 aligned sentence pairs that are shorter than 22 tokens (versus 633 sentence pairs in the original dataset, Table 1).

Table 5. Statistics for the dataset that considers sentence-length constraint. #Min-#Max specifies the range in "paragraph range" row. Also, 1-1 indicates the one-to-one mapping, 2-1 (1-2) indicates two-to-one and one-to-two mapping, and so on.

		all	dev	test
#input paragraphs		393	48	345
#input non-paraphrased pairs (dataset errors)		7	2	5
avg. paragraph length (#sentences)		2.3	2.4	2.3
avg. sentence length (#tokens)		20.9	19.3	21.1
paragraph range (# sentences)		1-7	1-6	1-7
% of alignment types	all	822 (100%)	87 (100%)	735 (100%)
	1-1 (all)	633 (77%)	67 (77%)	566 (77%)
	2-1 (1-2)	132 (16%)	16 (18%)	118 (16%)
	2-2	8 (1%)	1 (1%)	7 (1%)
	other (2-3,1-4,etc.)	49 (6%)	4 (4%)	45 (6%)
Evaluation Benchmark	**1-1 (<22 tokens, golden answers)**	**367 (45%)**	**50 (57%)**	**317 (43%)**

3.4.2.2 Experimental Results on Sentences with Limited Length

Tables 6-8 compare all similarity measures under the Best Match strategy (Uni- and Bi-directional, separately) and the Sequence Match strategy, respectively for the dataset containing only sentences shorter than 22 tokens. We follow the same scheme adopted in the previous experiments to report the new results.

Table 6. Alignment results on sentences shorter than 22 tokens for the uni-directional Best Match strategy. TH indicates the threshold value. The asterisk * marks the metrics that outperforms NGRAM baseline (n=3) with $p \leq 0.05$.

measure	% on the test set			Best *TH*
	prec	rec	F1	
NGRAM(n=1)*	73.3	77.9	75.5	0.3
NGRAM(n=2)*	73.3	77.9	75.5	0.3
NGRAM(n=3)	74.4	65.9	69.6	0.3
NGRAM(n=4)*	73.3	77.9	75.5	0.3
NGRAM(n=5)*	73.3	77.9	75.5	0.3
TOKENstring*	77.7	70.3	73.8	0.2
TOKENsyn	71.3	65.9	68.5	0.1
W2V	74.6	65.9	70.0	0.8
GLOVE	65.7	74.4	69.8	0.95
BERTword*	73.9	83.9	78.6	0.75
BERTcls	78.2	66.9	72.1	0.9
SBERTbert*	70.3	**90.2**	79.0	0.6
SBERTalbert	76.9	70.3	73.5	0.35
SBERTmini*	74.2	85.2	**79.3**	0.35
BERTprec*	77.6	70.0	74.1	0.9
BERTrec*	81.5	73.8	77.5	0.9
BERTf1*	**80.9**	72.2	76.3	0.9

Overall, the performances (in terms of F1 scores) on those length-limited sentences are lower than that on the full dataset (Table 2-4). The drop in F1 score ranges from 2.4% (bi-directional Best Match; 85.1% vs. 82.7%) to 6.1% (Sequence Match; 88.8% vs. 82.7%). One reason for causing the drops is that it implicitly removes the simplest alignment cases after filtering out those longer sentences, where the whole paragraph just consists of one single sentence. Another reason is that shorter sentences are easier to be mistakenly linked because they have less distinctive tokens. Detailed explanation will be delayed to the discussion section (Section 5).

Table 7. Alignment results on sentences shorter than 22 tokens for the bi-directional Best Match strategy. TH indicates the threshold value. The asterisk * marks the metrics that outperforms NGRAM baseline (n=3) with $p \leq 0.05$.

measure	% on the test set			Best *TH*
	prec	rec	F1	
NGRAM(n=1)	77.7	77.9	77.8	0.3
NGRAM(n=2)	77.7	77.9	77.8	0.3
NGRAM(n=3)	74.7	83.9	79.0	0.1
NGRAM(n=4)	77.7	77.9	77.8	0.3
NGRAM(n=5)	77.7	77.9	77.8	0.3
TOKENstring	79.4	70.3	74.6	0.2
TOKENsyn	73.6	76.3	74.9	0.05
W2V*	78.4	84.5	81.3	0.6
GLOVE	72.9	68.8	70.8	0.95
BERTword*	78.6	83.3	80.9	0.75
BERTcls	**83.7**	64.7	73.0	0.9
SBERTbert*	75.0	87.1	80.6	0.6
SBERTalbert*	77.1	**89.3**	**82.7**	0.25
SBERTmini*	76.0	89.0	82.0	0.25
BERTprec*	75.7	86.4	80.7	0.85
BERTrec*	76.2	82.0	81.3	0.85
BERTfl	81.8	72.2	76.7	0.9

Unlike in the experiments on the full dataset, two of the alignment strategies – Bi-directional Best Match and Sequence Match obtain the same F1 score (82.7%) with the SBERTalbert metric. This might indicate that the adjacency and dependency information used in Sequence Match (but not Best Match) is not as important for aligning sentences with limited length.

Table 8. Alignment results on sentences shorter than 22 tokens for the Sequence Match Best Match strategy. TH indicates the threshold value. The asterisk * marks the metrics that outperforms NGRAM baseline (n=3) with $p \leq 0.05$.

measure	% on the test set			Best *TH*
	prec	rec	F1	
NGRAM(n=1)	76.5	82.3	79.3	0.2
NGRAM(n=2)	76.5	82.3	79.3	0.2
NGRAM(n=3)	74.7	83.9	79.0	0.1
NGRAM(n=4)	76.5	82.3	79.3	0.2
NGRAM(n=5)	76.5	82.3	79.3	0.2
TOKENstring	76.5	83.3	79.8	0.15
TOKENsyn	73.4	78.2	75.7	0
W2V*	78.2	84.9	81.4	0.45
GLOVE	72.8	72.6	72.7	0.9
BERTword*	**78.6**	83.3	80.9	0.75
BERTcls	77.5	78.2	75.7	0.85
SBERTbert*	75.0	87.1	80.6	0.6
SBERTalbert*	77.1	89.3	**82.7**	0.25
SBERTmini*	76.0	89.0	82.0	0.25
BERTprec*	75.7	86.4	80.7	0.85
BERTrec	74.9	84.5	79.4	0.85
BERTf1*	76.1	**90.5**	**82.7**	0.85

Furthermore, just as on the full dataset, the best similarity measure varies under different search mechanisms. In the sequence match approach, two BERT-type measures (i.e., all SentenceBERT variants with the best being BERTalbert (82.7% F1 score)), and two of BERTScore variants (i.e., BERTprec with 80.7% F1 score and BERTf1 with 82.7% F1 score) and word2vec metric (i.e., W2V, 81.4% F1 score) significantly outperform the baseline. The SentenceBERT and BERTScore measure performs best, surpassing the character-trigram baseline method by 3.7% (82.7% vs. 79.0%).

Similarly, in the bi-directional best match approach, the best result is again obtained by the SentenceBERT measure (i.e., SBERTalbert) with the best F1-score of 82.7%, significantly

outperforming the character ngram similarity measure at 79.0%. This confirms the observation from previous experiments regarding the high suitability of sentence-embedding-based approaches in our task.

Last, in the uni-directional best match approach, several tested measures significantly outperform the baseline (69.6%), including SBERTmini (79.3%), SBERTbert (79.0%), BERTword (78.6%), BERTrec (77.5%), BERTf1(76.3%), NGRAM with n≠3 (75.5%), BERTprec (74.1%) and TOKENstring (73.8%). The measures that perform best in this search mechanism are again mostly those that encode the sentence as a whole, similar to other search mechanisms.

In comparison with the alignment results obtained from those sentences without length limitation, the F1-scores measures on length-limited sentences are lower (see the last item in Section 5). Although the performance of alignment of sentences with limited length is overall lower than on full data, we still prefer to impose the sentence length limitation, because it only slightly lowers the alignment performance but will offer considerable benefit while extracting the lexico-syntactic templates later.

3.4.3 Comparison of BERT Word-averaging and [CLS] Token Sentence Representation

Table 9. Comparison of results of BERT word-averaging and BERT [CLS] token-based similarity metrics on the full dataset. SM indicates Search Mechanism. The asterisk * indicates cases where the difference between two measures is statistically significant with p ≤ 0.05.

SM	measure	% on the test set		
		prec	rec	F1
Sequence Search	**BERTword**	91.5	82.2	**86.6**
	BERTcls	92.3	81.4	86.5
Best Match (uni)	**BERTword***	78.5	87.0	**82.5**
	BERTcls	81.9	67.9	74.3
Best Match (bi)	**BERTword***	82.3	86.4	**84.3**
	BERTcls	86.2	66.5	75.1

Comparing the performance of the methods using BERTword (i.e., word-averaging of BERT token embeddings) and the BERT [CLS] token, we observe that the BERTword achieves better performance regardless of the adopted search mechanism. Table 9 and Table 10 show how BERTword performs significantly better ($p<0.05$) than BERTcls regardless of the search mechanism for the dataset with sentence length constraint. The BERTword results are up to

6.5% higher, depending on the search mechanism (80.9% vs. 75.7% for sequence match; 80.9% vs. 73.0%, and 78.6% vs. 72.1% for bi- and uni-directional, respectively). For the full dataset, a noticeable difference can be observed for both versions of the Best Match approach with up to a 9.2% difference (84.3% vs. 75.1% and 82.5% vs. 74.3% for bi- and uni-directional, respectively). This is in line with the observation from Choi *et al.* (2021), who noted that interpreting the [CLS] token embedding as the sentence representation might be inferior to combining the individual sub-word embeddings obtained from BERT in some tasks.

Table 10. Comparison of results of BERT word-averaging and BERT [CLS] token-based similarity metrics on sentences shorter than 22 tokens. SM indicates Search Mechanism. The asterisk * indicates cases where the difference between two measures is statistically significant with $p \leq 0.5$.

SM	measure	% on the test set		
		prec	rec	F1
Sequence Search	**BERTword***	78.6	83.3	**78.6**
	BERTcls	77.5	78.2	77.5
Best Match (uni)	**BERTword***	73.9	83.9	**78.6**
	BERTcls	78.2	66.9	72.1
Best Match (bi)	**BERTword***	78.6	83.3	**80.9**
	BERTcls	83.7	64.7	73.0

3.5 Exploring Features for Non-paraphrased Paragraph-pair Detection

As shown in Table 1, we have found that some of the paragraph pairs we sampled from the Webis-CPC-11 were mislabeled as paraphrase-pairs, in which the meaning of the two paragraphs is not similar. Figure 5 shows an example of such a non-paraphrased pair, where the text fragments in red indicate two different meanings. In one paragraph the character “Sukey” is said to have heard about some issues, whereas in the other paragraph it is indicated she has no idea about them. In the 400 pairs with the positive labels that we sampled, 7 were not paraphrases.

Although we have excluded those outlier pairs from our previous experiments, they are manually detected, which would be too time-consuming to do so for a large corpus. Therefore, we would like to check whether it is possible to detect such incorrectly labeled data automatically. As the paragraph is just a longer passage in comparison with the sentence, we expect that the measures adopted to calculate the sentence similarity could be also applied to evaluate the paragraph similarity. We thus further test whether the measures adopted for sentence alignment are discriminative enough to filter out those incorrectly annotated paragraph

pairs (i.e., non-paraphrased pairs found).

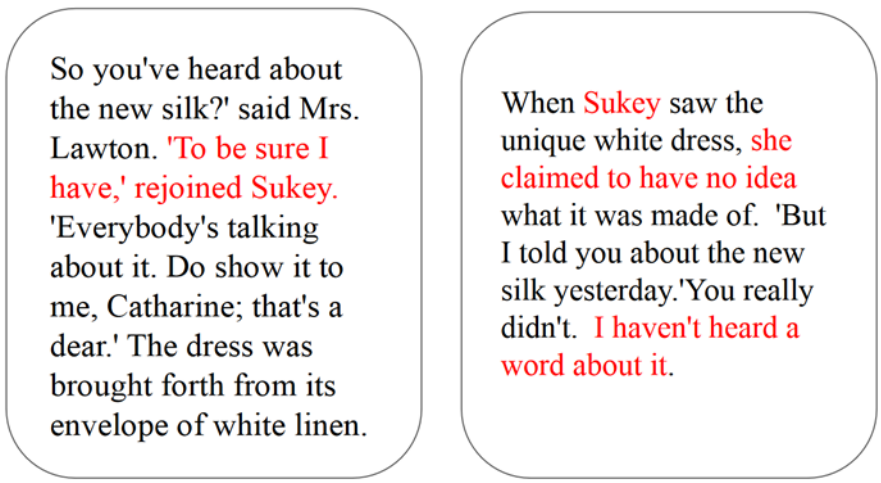

Figure 5. Example of a non-paraphrased paragraph pair (outlier) from the Webis-CPC-11 dataset. Red marks text fragments with opposite meanings.

To detect the outliers, we first calculate the paragraph similarity using the same similarity measures adopted in the previous experiments, but taking paragraphs, not sentences, as the input. We include the following similarity measures in the experiment: (1) based on the unit-overlap-ratio (including: NGRAM(n=3), TOKENstring, TOKENsyn); based on the sentence-vector-similarity (including SentenceBERT and BERTScore). We model the similarity values from all paraphrased paragraph pairs for each measure with a specific normal distribution and then calculate its 0.95 confidence interval to check whether the non-paraphrased paragraphs can be detected as outliers outside this interval.

Table 11 shows the percentage of non-paraphrased pairs that fall below the left boundary value of the 0.95 Confidence Interval for each of the adopted similarity measures. The best result is achieved using BERTprec, with which we can detect all outlier pairs. This leads to the conclusion that it is possible to automatically detect those non-paraphrased paragraph-pairs by using BERTScore as a similarity measure.

Table 11. Results of filtering out non-paraphrased paragraph pairs based on the 0.95 confidence interval. Mean is the mean similarity value for all (393) paraphrased paragraph pairs; L-CI is the left boundary of the Confidence Interval, and #pairs is the number of non-paraphrased pairs that fall outside the confidence interval (out of 7). Results with $p \leq 0.05$ are marked with the asterisk *. Table adopted from Smolka et al. (2022).

measure	mean	L-CI (0.95)	% pairs
NGRAM(n=3)	0.547	0.530	71%
TOKENstring	0.221	0.214	57%
TOKENsyn	0.141	0.136	57%
SBERTbert	0.541	0.522	43%
SBERTalbert	0.411	0.391	43%
SBERTmini*	0.339	0.321	86%
BERTprec*	0.914	0.911	**100%**
BERTrec	0.917	0.914	71%
BERTf1*	0.915	0.913	71%

4. Error Analysis

We analyzed 50 errors generated by our best approach (i.e., Sequence Match with SBERTmini), and categorized them based on their associated error sources: (1) mistaking 1-n mapping for 1-1 (46%); (2) associated with incorrect sentence boundary (26%), in which the sentences are split incorrectly before conducting alignment (e.g., a sentence is incorrectly split into two sequences by the sentence segmenter); (3) paraphrased sentences take different sequence-orders within two given paragraphs (16%); (4) others (12%), of which it is difficult to attribute each error to a specific reason.

Table 12 shows an example of the first error category, which incorrectly marks a 1-n alignment as 1-1. The source of this error is likely due to the following two reasons. First, those proposed similarity measures are still incapable of truly reflecting the semantic similarity between two sentences when they are paraphrased in an abstract way; as a result, they might incorrectly convert a golden 1-n mapping into a 1-1 mapping. Second, because the alignment is selected based on the sentence similarity and the probability of each alignment type is estimated from the development set, the adopted model has a preference for extracting 1-1 alignments as they are most common in the dataset (cf. Table 1).

Table 12. An example which mis-interprets a one-to-many relationship as a 1-1 alignment. Gold sentence alignments (i.e., pairs "a", "b") are correctly extracted; "x" is incorrectly extracted and "0" is an annotated 1-n alignment which we do not want to extract.

		PARAGRAPH #1	PARAGRAPH #2
MODEL INPUT (FULL PARAGRAPHS)		**Thad, of course. And, Bill, we're going to get him, sooner or later.** Mr. Hooper won't want to stand this sort of thing forever. I've got a hunch that we're not through with that game yet.	**Naturally, Thad and also Bill, whom we'll get after a while.** Mr. Hooper won't let this go on for long. I'm guessing we won't be done for some time.
ALIGNED SENTENCES (GOLDEN ANSWER)	0	Thad, of course. And, Bill, we're going to get him, sooner or later.	Naturally, Thad and also Bill, whom we'll get after a while
	a	Mr. Hooper won't want to stand this sort of thing forever.	Mr. Hooper won't let this go on for long.
	b	I've got a hunch that we're not through with that game yet.	I'm guessing we won't be done for some time.
MODEL ANSWER	x	And, Bill, we're going to get him, sooner or later.	Naturally, Thad and also Bill, whom we'll get after a while.
	a	Mr. Hooper won't want to stand this sort of thing forever.	Mr. Hooper won't let this go on for long.
	b	I've got a hunch that we're not through with that game yet.	I'm guessing we won't be done for some time.

The second error category (i.e., with incorrect sentence boundary) occurs when the pre-processing module incorrectly split the sentences within one of the input paragraphs. Finally, the last type of error is caused by the sequence search mechanism, which assumes all paraphrased passage pairs follow the same relative order within each paragraph. If this assumption is violated in the given paragraph pair, it will always return an incorrect answer.

5. Discussion

Based on our results, we get the following observations:

- Among various sentence alignment strategies, Sequence Match tends to give the best and most consistent results across all our experiments. The advantage of Sequence Match is that it employs dynamic programming which makes it faster than the greedy approaches. It also

performs well where the adjacency and dependency information between sentences is relevant to the matching. However, it will not perform well when the sentences are in a different order in the two paragraphs, in which case using the Best Match strategy would be preferable. Furthermore, the bi-directional Best Match shows much better performance than the uni-directional approach on both datasets we use, which can be explained by the symmetry in our data, as described earlier in the introduction section.

- In general, the measures that encode the sentence directly tend to perform better than those that are based on individual token representations (either unit-overlap or token-embedding-average). The only exception is the approach using the BERT [CLS] token. We believe it might be because the [CLS] token is not explicitly trained to condense a long text sequence into a vector, unlike SentenceBERT and BERTScore which are created specifically for doing so.
- The method using averaged word vectors from BERT outperforms the method using the [CLS] token in our task. The inferior performance of the method with [CLS] token representations might be due to that the [CLS] token is trained on a much smaller amount of data; in contrast, those individual token embeddings are trained from a much larger dataset.
- Noticeably, the best thresholds of those non-embedding methods tend to be much lower than those of the measures that utilize neural embeddings. We conjecture this is because the neural models estimate similarity based on soft/fuzzy matching (which would result lower thresholds), while string-based methods use hard/strict matching (which would result higher thresholds, as it cannot distinguish the soft matching case from the un-matched case).
- Finally, we have discovered that when the additional sentence length limitation is imposed, the performance drops across all approaches, with the biggest difference for the Sequence Matching approach. One possible explanation is that shorter sentences are easier to be mistakenly linked because they have less distinctive tokens (e.g., when comparing short sentences like "*John Walker went*." and "*John Walker came*.", the similarity between them will be always high because there is only one distinguishing token; however, it would be a less serious issue for the cases with longer sentences). Another reason might be that the sentence length limitation implicitly removes the trivial cases from the dataset, i.e., those cases where the whole paragraph only contains a single long sentence that will be automatically mapped to its corresponding paragraph (and forms a 1-1 mapping). Such cases are more likely to appear in the full dataset, which would make the overall result higher on this dataset.

6. Conclusions

We have presented the first comparison among various model-agnostic similarity measures used for aligning sentences among paraphrased paragraphs. For most cases, we find that embedding-based similarity measures outperform the string-based approaches (including the previous

SOTA character trigram approach tested on the TS dataset), and sentence-embedding-based methods are preferable to the word-embedding-based methods for most search mechanisms except the uni-directional greedy matching.

Additionally, our results have shown that in calculating the similarity for sentence alignment, word vector averaging is better than adopting the [CLS] token when retrieving a representation of a whole sentence from a BERT-based model.

References

Aghaebrahimian, A. (2017). Quora Question Answer Dataset. *TSD. Text, Speech, and Dialogue*, 66-73. https://doi.org/10.1007/978-3-319-64206-2_8

Barzilay, R., & McKeown, K. (2001). Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics (ACL '01)*, 50-57. https://doi.org/10.3115/1073012.1073020

Barzilay, R., & McKeown, K. (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics*, *31*(3), 297-328. https://doi.org/10.1162/089120105774321091

Blevins, T., Levy, O., & Zettlemoyer, L. (2018). Deep RNNs Encode Soft Hierarchical Syntax. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 14-19. https://doi.org/10.18653/v1/P18-2003

Burrows, S., Potthast, M., & Stein, B. (2013). Paraphrase acquisition via crowdsourcing and machine learning. *Transactions on Intelligent Systems and Technology (ACM TIST)*, *4*(3), 1-21. https://doi.org/10.1145/2483669.2483676

Choi, H., Kim, J., Joe, S., & Gwon, Y. (2021). Evaluation of BERT and ALBERT Sentence Embedding Performance on Downstream NLP Tasks. In *Proceedings of the 25th International Conference on Pattern Recognition (ICPR 2020)*, 5482-5487 https://doi.org/10.1109/ICPR48806.2021.9412102

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171-4186. https://doi.org/10.18653/v1/N19-1423

Dietterich, T.G. (1998). Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, *10*(7), 1895-1923. https://doi.org/10.1162/089976698300017197

Dolan, W.B., & Brockett, C. (2005). Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Fellbaum, Ch., (1998). *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

Gale, W.A., & Church, K.W. (1993). A Program for Aligning Sentences in Bilingual Corpora. *Computational Linguistics*, *19*(1), 75-102.

Ganitkevitch, J., Durme, B.V., & Callison-Burch, C. (2013). PPDB: The Paraphrase Database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 758-764.

Hupkes, D., Dankers, V., Mul, M., & Bruni, E. (2020). Compositionality Decomposed: How do Neural Networks Generalise? (Extended Abstract). In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, 5065-5069. https://doi.org/10.24963/ijcai.2020/708

Jiang, C., Maddela, M., Lan, W., Zhong, Y., & Xu, W. (2020). Neural CRF Model for Sentence Alignment in Text Simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online. Association for Computational Linguistics*, 7943-7960. http://dx.doi.org/10.18653/v1/2020.acl-main.709

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2020). ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *Proceedings of 8th International Conference on Learning Representations (ICLR 2020)*. https://doi.org/10.48550/arXiv.1909.11942

Liu,Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. Computing Research Repository, arXiv:1907.11692. Version 1. https://doi.org/10.48550/arXiv.1907.11692

MacCartney, B., & Manning, C.D. (2008). Modeling Semantic Containment and Exclusion in Natural Language Inference. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, 521-528.

McNamee, P., & Mayfield, J. (2004). Character N-Gram Tokenization for European Language Text Retrieval. *Information Retrieval*, 7, 73-97. https://doi.org/10.1023/B:INRT.0000009441.78971.be

McCoy, T., Pavlick, E., & Linzen, T. (2019). Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 3428-3448. https://doi.org/10.18653/v1/P19-1334

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. In *Proceedings of 1st International Conference on Learning Representations (ICLR 2013)*. https://doi.org/10.48550/arXiv.1301.3781

Pennington, J., Socher, R., & Manning, C. (2014). GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532-1543. https://doi.org/10.3115/v1/D14-1162

Putra, J.W.G., & Tokunaga, T. (2017). Evaluating text coherence based on semantic similarity graph. In *Proceedings of TextGraphs-11: the Workshop on Graph-based Methods for Natural Language Processing*, 76-85. https://doi.org/10.18653/v1/W17-2410

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 3982-3992. https://doi.org/10.18653/v1/D19-1410

Rosenblatt, M. (1956). Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics*, *27* (3). 832-837. https://doi.org/10.1214/aoms/1177728190

Smolka, A., Wang, H.M., Chang, J.S., & Su, K.Y. (2022). Is Character Trigram Overlapping Ratio Still the Best Similarity Measure for Aligning Sentences in a Paraphrased Corpus? In *Proceedings of the 34th Conference on Computational Linguistics and Speech Processing (ROCLING 2022)*, 49-60.

Štajner, S., Franco-Salvador, M., Rosso, P., & Ponzetto, S.P. (2018). CATS: A Tool for Customized Alignment of Text Simplification Corpora. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Štajner, S., Franco-Salvador, M., Rosso, P., Ponzetto, S.P., & Stuckenschmidt, H. (2017). Sentence Alignment Methods for Improving Text Simplification Systems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 97-102. https://doi.org/10.18653/v1/P17-2016

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., & Manning, C.D. (2020). Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 101-108. https://doi.org/10.18653/v1/2020.acl-demos.14

Qiu, L., Kan, M., & Chua, T. (2006). Paraphrase Recognition via Dissimilarity Significance Classification. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, 18-26.

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS'20)*, 5776-5788. https://dl.acm.org/doi/abs/10.5555/3495724.3496209

Weiss, D., Roit, P., Klein, A., Ernst, O., & Dagan, I. (2021). QA-Align: Representing Cross-Text Content Overlap by Aligning Question-Answer Propositions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 9879-9894. https://doi.org/10.18653/v1/2021.emnlp-main.778

Zhou, J., Zhang, Z., Zhao, H. & Zhang, S. (2020). LIMIT-BERT : Linguistics Informed Multi-Task BERT. In *Proceedings of Findings of the Association for Computational Linguistics: EMNLP 2020*, 4450-4461. https://doi.org/10.18653/v1/2020.findings-emnlp.399