Compositional Phrase Alignment and Beyond

Yuki Arase[1⋆] and **Junichi Tsujii**[⋆2]
[1]Osaka University, Japan
[⋆]Artificial Intelligence Research Center (AIRC), AIST, Japan
[2]NaCTeM, School of Computer Science, University of Manchester, UK
arase@ist.osaka-u.ac.jp, j-tsujii@aist.go.jp

Abstract

Phrase alignment is the basis for modelling sentence pair interactions, such as paraphrase and textual entailment recognition. Most phrase alignments are compositional processes such that an alignment of a phrase pair is constructed based on the alignments of their child phrases. Nonetheless, studies have revealed that non-compositional alignments involving long-distance phrase reordering are prevalent in practice. We address the phrase alignment problem by combining an unordered tree mapping algorithm and phrase representation modelling that explicitly embeds the similarity distribution in the sentences onto powerful contextualized representations. Experimental results demonstrate that our method effectively handles compositional and non-compositional global phrase alignments. Our method significantly outperforms that used in a previous study and achieves a performance competitive with that of experienced human annotators.

1 Introduction

Phrase alignment is a fundamental problem in modelling the interactions between a pair of sentences, such as paraphrase identification, textual entailment recognition, and question answering (Das and Smith, 2009; Heilman and Smith, 2010; Wang and Manning, 2010). Phrase alignment generally adheres to compositionality, in which a phrase pair is aligned based on the alignments of their child phrases. Nonetheless, non-compositional alignments involving long-distance phrase reordering are prevalent in practice (Burkett et al., 2010; Heilman and Smith, 2010; Arase and Tsujii, 2017). Figure 1 shows an example of phrase alignment in which phrases of the same colours are alignable, *i.e.* they are phrasal paraphrases. The alignment of 'antivirus vaccines' and 'vaccines against the virus' is compositional, as supported by alignments of their child nodes although their orderings are reversed. Similarly, the alignment of their parents τ_2^s and τ_2^t is compositional. By contrast, the alignment of τ_1^s and τ_1^t is non-compositional in relation to the alignment of τ_2^s and τ_2^t; although τ_1^t and τ_2^t are siblings, τ_1^s is not a sibling of τ_2^s, *i.e.* not in the scope of the parent node of τ_2^s. To treat such a long-distance correspondence in non-compositional alignment, one has to consider candidate phrases outside the local scope and potentially the entire sentence.

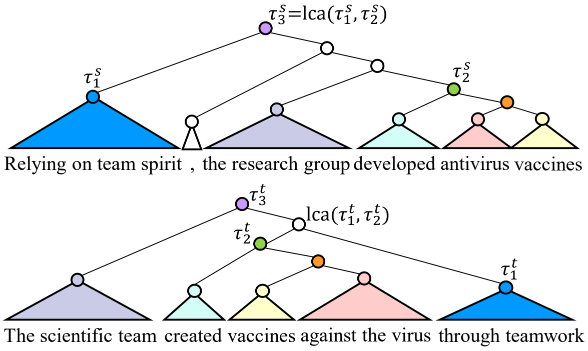
τ_3^s=lca(τ_1^s, τ_2^s)
τ_1^s
τ_2^s
Relying on team spirit , the research group developed antivirus vaccines
τ_3^t
lca(τ_1^t, τ_2^t)
τ_2^t
τ_1^t
The scientific team created vaccines against the virus through teamwork

Figure 1: Phrase alignments by the proposed method (phrases of the same colour are paraphrases)

In this study, we address the phrase alignment problem by combining a tree mapping algorithm with phrase representation modelling. We treat compositional alignment by an algorithm for an *unordered* tree mapping (Zhang, 1996). For the algorithm to work, definition of the edit cost (*i.e.* dissimilarity between phrases) is crucial. We propose a novel phrase representation, by which the edit cost is defined, based on contextualized representations by the bidirectional encoder representations from transformers (BERT) (Devlin et al., 2019). The proposed phrase representation models the similarity distribution in the entire sentence, thereby allowing the algorithm to be extended to treat non-compositional global alignments.

Phrase alignment can be difficult even for humans because there is unavoidable subjectivity in

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 1611–1623,
November 16–20, 2020. ©2020 Association for Computational Linguistics

acceptable semantic discrepancies between paraphrases. Our experimental results indicate that the proposed method achieves 95.7% of the alignment quality of trained human annotators for phrase alignment in paraphrase sentence pairs.

The contributions of this study are twofold. First, we formalise the compositional phrase alignment problem as an unordered tree mapping. Second, we propose a phrase representation model that allows non-compositional global alignments.

2 Related Work

2.1 Tree Mapping and Phrase Alignment

Ordered tree mapping has been employed to estimate the similarity of a pair of sentences for its ability to align syntactic trees (Punyakanok et al., 2004; Alabbas and Ramsay, 2013; Yao et al., 2013; McCaffery and Nederhof, 2016). However, it is too restrictive in that the order of the aligned phrases in the sentences must be the same. Previous studies extended the algorithm to adapt the edit costs (Bernard et al., 2008; Mehdad, 2009; Alabbas and Ramsay, 2013) and edit operations (Heilman and Smith, 2010; Wang and Manning, 2010) to specific tasks. In contrast, the unordered tree mapping that we employ in this study is sufficiently flexible to assure identification of optimal compositional phrase alignments.

Parallel parsing also involves phrase alignment in its parsing process. As the tree isomorphism assumption is too restrictive, previous studies have employed various relaxation techniques that prefer but do not force synchronisation. Burkett et al. (2010) used weakly synchronised grammar, and Das and Smith (2009) used quasi-synchronous grammars (Smith and Eisner, 2006). Choe and McClosky (2015) used dual decomposition to encourage agreement between two parse trees. All of these methods allow excess flexibility beyond compositionality in alignment. Rule extraction for tree transducers also involves phrase alignments (Martínez-Gómez and Miyao, 2016) but disregards phrase boundaries to maximise the coverage of extracted rules. In contrast, the phrase alignment problem addressed in our study adheres to syntactic structures.

2.2 Phrase Representation Generation

Researchers have proposed specialised phrase representations for specific tasks (Arase and Tsujii, 2019; Yin et al., 2020) on top of contextualised representations. In this study, we propose dedicated phrase representations for the alignment problem. Before contextualised representation, studies considered word alignment distributions for modelling semantic interactions between a pair of sentences (He and Lin, 2016; Parikh et al., 2016; Chen et al., 2017). We agree with their intuition that the pairwise similarities alone are not good enough to define the cost of alignment. In case there are other similar phrases, their pairwise similarities have to be properly adjusted. This adjustment is crucial for treating non-compositional global alignment.

3 Phrase Alignment Method

3.1 Preliminaries and Notation

We refer to one of the paraphrasal sentences as the *source*, s, and the other as the *target*, t. Superscripts s and t represent source and target, respectively. The syntactic trees of the source and target, $T^s = \{\tau_i^s\}_i$ and $T^t = \{\tau_j^t\}_j$, determine the phrase structures; τ_i^s and τ_j^t are the source and target phrases. The alignments of their phrases are $\mathbb{H} = \{\mathbb{h}_i = \langle \tau_i^s, \tau_i^t \rangle\}_i$. We interchangeably use the subscript of a node as the index of the alignment or the index of the node in a tree whenever the meaning is apparent from the context. A phrase can align to an empty node $\tau_\emptyset$ ($\tau_\emptyset \notin T$), which is called the *null* alignment.

We define functions to traverse a tree: $\mathrm{ds}(\tau)$ derives descendant nodes of τ, and $\mathrm{lca}(\tau_i, \tau_j)$ derives the lowest common ancestor of τ_i and τ_j. Additionally, function $\mathrm{deg}(T)$ computes the maximum depth of T, and $|\cdot|$ counts the number of elements in a set; *e.g.* $|T|$ is the number of nodes in T.

3.2 Problem Definition

Based on Arase and Tsujii (2017), we reformalise conditions of legitimacy as a set of compositional phrase alignments $\mathbb{H}_L$.

Definition 3.1. *Legitimacy conditions consist of the following:*

Consistency *In $\mathbb{H}_L$, a phrase ($\neq \tau_\emptyset$) in the source tree is aligned with at most one phrase ($\neq \tau_\emptyset$) in the target tree, and vice versa.*

Monotonicity *For $\langle \tau_i^s, \tau_i^t \rangle, \langle \tau_j^s, \tau_j^t \rangle \in \mathbb{H}_L$, $\tau_i^s \in ds(\tau_j^s)$ iff $\tau_i^t \in ds(\tau_j^t)$.*

Familiness *For $\langle \tau_1^s, \tau_1^t \rangle$, $\langle \tau_2^s, \tau_2^t \rangle$, $\langle \tau_3^s, \tau_3^t \rangle$ in $\mathbb{H}_L$,*

$lca(\tau_1^s, \tau_2^s)$ is a proper ancestor[1] *of τ_3^s iff $lca(\tau_1^t, \tau_2^t)$ is a proper ancestor of τ_3^t.*

The consistency condition ensures one-to-one alignment. The monotonicity condition regulates the retainment of the ancestor-descendant relation in the source and target sides. The familiness condition realises compositionality in the language, which constrains such that two separate subtrees of T^s should be aligned to two separate subtrees of T^t.[2] In other words, the familiness condition prohibits a node in the source subtree to align to a node outside that target subtree. In Figure 1, $\langle \tau_1^s, \tau_1^t \rangle$ violates the familiness condition in relation to $\langle \tau_2^s, \tau_2^t \rangle$ and $\langle \tau_3^s, \tau_3^t \rangle$ because τ_3^s is not a proper ancestor of $\mathrm{lca}(\tau_1^s, \tau_2^s)$, whereas τ_3^t is a proper ancestor of $\mathrm{lca}(\tau_1^t, \tau_2^t)$.

We define non-compositional alignments $\mathbb{H}_{\mathrm{nc}}$ as alignments that satisfy the legitimacy conditions internally but do not satisfy them against $\mathbb{H}_L$. For example, the alignment $\langle \tau_1^s, \tau_1^t \rangle$ in Figure 1 is compositionally composed and satisfies the legitimacy conditions for its internal alignments. However, it does not satisfy the legitimacy conditions against alignments of $\langle \tau_2^s, \tau_2^t \rangle$ and $\langle \tau_3^s, \tau_3^t \rangle$ for violation of the familinesss condition. We allow $\mathbb{H}_{\mathrm{nc}}$ to be added into $\mathbb{H}_L$ if it is compatible;

Definition 3.2. *$\mathbb{H}_{nc}$ is compatible with $\mathbb{H}_L$ iff for all $\langle \tau_i^s, \tau_i^t \rangle \in \mathbb{H}_{nc}$ ($\tau_i^s, \tau_i^t \neq \tau_\emptyset$), both $\langle \tau_i^s, \tau_\emptyset \rangle$ and $\langle \tau_\emptyset, \tau_i^t \rangle$ are in $\mathbb{H}_L$.*

When the compatibility condition is met, $\mathbb{H}_{\mathrm{nc}}$ can be safely added to $\mathbb{H}_L$ by complementing null alignments without violating the consistency condition. We implement this process by a simple post-processing step (Section 3.4).

3.3 Compositional Alignment

Finding the optimal set of legitimate compositional alignments (Definition 3.1) is equivalent to finding the minimum cost of constrained tree mapping (Zhang, 1996), which belongs to the problem of *unordered* tree mapping (Bille, 2005). The edit operations of re-labelling, deletion, and insertion correspond to alignment of two nodes, null alignment of a source node, and null alignment of a target node, respectively. Although the unordered tree mapping problem is in general MAX SNP-hard (Zhang and Jiang, 1994), the constrained tree edit distance (CTED) algorithm (Zhang, 1996) achieves polynomial time complexity using the familiness condition. In essence, the CTED algorithm reduces the unordered tree mapping problem to a maximum matching problem by the familiness condition. The reduction enables faster dynamic programming of $\mathcal{O}(|T^s||T^t|(\deg(T^s) + \deg(T^t)) \log(\deg(T^s) + \deg(T^t)))$. Details of the CTED algorithm are described in detail in Appendix B.

To apply CTED for phrase alignment, the edit cost function $\gamma(\cdot) \rightarrow \mathbb{R}$ is the key, which should satisfy the properties of a proper distance metric. This function evaluates the *dissimilarity* of a phrase pair, for which we propose a phrase representation model (Section 4). We use cosine distance as $\gamma(\cdot) \in [0, 2.0]$ because of its prevalence in measuring dissimilarity between representations. However, it is not a proper distance metric because it does not satisfy the triangle inequality property. In future work, we will investigate alternative distance metrics.

We also need to estimate the cost of a null alignment. It is not trivial to generate representation of such an empty phrase; hence, we decided to use a constant cost $\lambda_\emptyset$, *i.e.*,

$$\gamma(\langle \tau^s, \tau_\emptyset \rangle) = \gamma(\langle \tau_\emptyset, \tau^t \rangle) = \lambda_\emptyset \in [0, 2.0].$$

The appropriate value of $\lambda_\emptyset$ is determined using a development set.

3.4 Non-compositional Alignment

We designed top-down post-processing for non-compositional alignment so that the legitimacy conditions (Definition 3.1) will be maximally satisfied in the final alignments. As Algorithm 3.1 shows, we add a set of alignments $\mathbb{H}_{\mathrm{nc}}$ that compose the non-compositional alignments into $\mathbb{H}_L$ when they are compatible. Our post-processing aligns all the coloured phrase pairs in Figure 1 by allowing $\langle \tau_1^s, \tau_1^t \rangle$ and its descendant alignments.

Algorithm 3.1 takes matrices of edit distance and corresponding operations D and A as input, which are obtained by CTED. $D[i+1][j+1]$ and $A[i+1][j+1]$ store the total cost and operations, respectively, to compose alignment of $\langle \tau_i^s, \tau_j^t \rangle$. Note that index 0 is reserved for null alignments. The algorithm sorts null alignments in $\mathbb{H}_L$ in descending order of the span covering the source and target phrases (line 2). For each null alignment, the algorithm finds candidates of non-compositional

[1] A proper ancestor of a node i is any node j such that node j is an ancestor of node i and j is not the same node as i.

[2] Our definition is less constrained than that in Arase and Tsujii (2017) as discussed in Appendix A.

Algorithm 3.1 Non-compositional alignment

```
Input: Legitimate alignments $\mathbb{H}_L$ and matrices of tree edit distance and corresponding operations $D$ and $A$
 1: $\mathbb{H}_\emptyset \leftarrow \{\langle \tau^s, \tau_\emptyset \rangle, \langle \tau_\emptyset, \tau^t \rangle | \mathbb{H}_L\}$
 2: Sort $\mathbb{H}_\emptyset$ by descending order of phrase span
 3: for all $\langle \tau_i^s, \tau_j^t \rangle \in \mathbb{H}_\emptyset$ do
 4:   if $\tau_j^t = \tau_\emptyset$ then                ▷ target side is $\tau_\emptyset$
 5:     for all $k \in \operatorname*{argmin}_\ell D[i+1][\ell]$ do
 6:       if IsCompatible($A[i+1][k], \mathbb{H}_L$) then UpdateAlignments($A[i+1][k], \mathbb{H}_L, \mathbb{H}_\emptyset$)
 7:   else
 8:     Do the same for the source side
 9: function IsCompatible($\hat{A}$, $\mathbb{H}_L$)
10:   for all $\langle \tau_i^s, \tau_j^t \rangle \in \hat{A}$ where $\tau_i^s, \tau_j^t \neq \tau_\emptyset$ do
11:     if $\langle \tau_i^s, \tau_k^t \rangle \in \mathbb{H}_L$ or $\langle \tau_l^s, \tau_j^t \rangle \in \mathbb{H}_L$ where $\tau_k^t, \tau_l^s \neq \tau_\emptyset$ then return False
12:   return True
13: function UpdateAlignments($\hat{A}$, $\mathbb{H}_L$, $\mathbb{H}_\emptyset$)
14:   for all $\langle \tau_i^s, \tau_j^t \rangle \in \hat{A}$ where $\tau_i^s, \tau_j^t \neq \tau_\emptyset$ do
15:     $\mathbb{H}_L \leftarrow \mathbb{H}_L \cup \{\langle \tau_i^s, \tau_j^t \rangle\}$
16:     Remove $\langle \tau_i^s, \tau_\emptyset \rangle$ from $\mathbb{H}_L$ and $\mathbb{H}_\emptyset$
17:     Remove $\langle \tau_\emptyset, \tau_j^t \rangle$ from $\mathbb{H}_L$ and $\mathbb{H}_\emptyset$
```

alignments achieving the minimum cost (line 5). Then, using the IsCompatible function, it checks whether a non-compositional alignment and its descendant alignments are compatible with the current set of alignments. If so, they are added to $\mathbb{H}_L$ by the UpdateAlignments function, replacing null alignments $\langle \tau_i^s, \tau_\emptyset \rangle$ and $\langle \tau_\emptyset, \tau_j^t \rangle$ in $\mathbb{H}_L$ with non-compositional alignment $\langle \tau_i^s, \tau_j^t \rangle$.

Our post-processing is a heuristic to maximally satisfy the legitimacy conditions, as finding the best combination of non-compositional alignments is computationally intractable.[3] Our method ensures that non-compositional alignments improve the alignment cost by only allowing those with minimum cost.

4 Phrase Representation for Alignment

We propose a phrase representation model on top of the pre-trained BERT. One of the most common methods for obtaining a phrase representation from BERT is pooling outputs corresponding to tokens in the phrase. However, as we empirically show in Section 6, this method exhibits an unsatisfactory ability for modelling the similarity distribution in a sentence pair. Hence, we propose a novel method for generating phrase representations suitable for the phrase alignment problem.

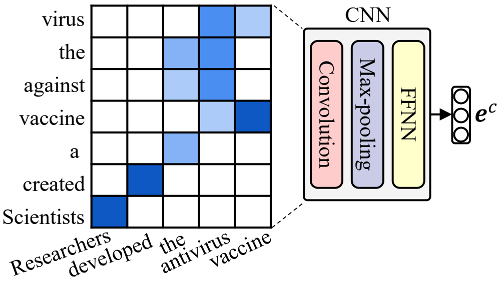

Figure 2: Modelling similarity distribution (shades of the matrix represent word similarities)

Problem Statement and Approach The estimate of a phrase pair's similarity for alignment is unique, because their similarity should depend on similarities of other phrases in the sentence pair. That is, even if the pairwise similarity of τ_i^s and τ_j^t is high, the similarity score should be lowered if there is a phrase in the source sentence that is more similar to τ_j^t. Hence, we generate a phrase representation that reflects the similarity distribution within the sentence pair; this is particularly important for non-compositional alignments to find a globally plausible alignment pair.

We first generate a representation of the similarity distribution within the sentence pair. We then transform the phrase representation obtained from BERT, referring to the representation of the similarity distribution using an attention mechanism.

Similarity Distribution Modelling We regard outputs of the last layer $\boldsymbol{h} \in \mathbb{R}^b$ of BERT as token representations, where b is the hidden size determined by the BERT pre-training settings. Using the token representations, we generate a representation of similarity distribution $\boldsymbol{e}^c \in \mathbb{R}^b$ (Figure 2).

We first compute cosine similarities between token representations of the sentence pair and obtain the similarity matrix. We then encode the similarity matrix using a convolutional neural network (CNN) and obtain $\boldsymbol{e}^c$, called the *SimMatrix* representation. Our CNN is shallow, under the assumption that a shallow model is sufficient to capture latent features in SimMatrix. A shallow model also allows training with a smaller corpus while fine-tuning BERT. The CNN consists of a one-channel convolution layer activated by the rectified linear unit function, a max-pooling layer, and a fully connected feed-

[3] Arase and Tsujii (2017) do not assure maximal satisfaction of the legitimacy conditions.

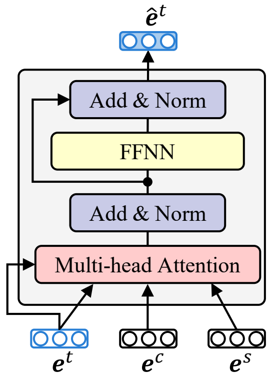

Figure 3: Phrase representation transformation

forward neural network (FFNN).

Representation Generation We obtain a basic representation of τ^s for span i to j by simply pooling the token representations obtained from BERT: $\boldsymbol{e}^s = \text{pool}(\boldsymbol{h}_i, \ldots, \boldsymbol{h}_j) \in \mathbb{R}^b$. Similarly, a basic representation $\boldsymbol{e}^t$ of target phrase τ^t is obtained. We then transform $\boldsymbol{e}^t$ to reflect the SimMatrix representation $\boldsymbol{e}^c$. For this, we use an attention mechanism as shown in Figure 3, which has the same architecture as the Transformer (Vaswani et al., 2017). The attention layer consists of multi-head attention and FFNNs. Our model takes $\boldsymbol{e}^c$, $\boldsymbol{e}^s$, and $\boldsymbol{e}^t$, and transforms $\boldsymbol{e}^t$ into $\hat{\boldsymbol{e}}^t \in \mathbb{R}^b$.

Loss Function To train the phrase representation model, we use a triplet margin loss:

$$\mathcal{L}(\boldsymbol{e}^s, \hat{\boldsymbol{e}}^t_p, \hat{\boldsymbol{e}}^t_n) = \max\{\|\boldsymbol{e}^s - \hat{\boldsymbol{e}}^t_p\|_2 - \|\boldsymbol{e}^s - \hat{\boldsymbol{e}}^t_n\|_2 + \delta, 0\}, \quad (1)$$

where $\hat{\boldsymbol{e}}^t_p$ and $\hat{\boldsymbol{e}}^t_n$ are transformed representations of positive (alignable) and negative (unalignable) pairs, respectively, and δ is a margin. Intuitively, the loss function makes representations of paraphrase pairs closer, whereas those of non-paraphrase pairs are more distant. For negative examples, we randomly sample phrases that are separated by more than one hop from the alignable pair in T^t. At an inference, we transform the basic representation of a target phrase by our model and compute the cost $\gamma(\boldsymbol{e}^s, \hat{\boldsymbol{e}}^t)$.

We also tried models that discriminate alignable phrases or minimise the cosine similarity of an alignable pair. However, they were all inferior to the triplet margin loss.

	ESPADA	SPADE	
		dev	test
# sentence pairs	1,916	50	151
# phrases w/o tokens	75,283	2,584	7,438
Total # pairs	251,972	8,708	25,709
# unique pairs	105,154	3,566	10,790
# pairs agreed by ≥ 2	80,572	2,814	8,292
# pairs agreed by all	66,246	2,328	6,627
Non-monotonicity	3.6%	4.7%	3.2%
Non-familiness	1.4%	1.2%	1.1%

Table 1: Statistics for ESPADA and SPADE ('#' stands for 'number of')

5 Experiment Setting

5.1 Creation of ESPADA

To train our phrase representation model, we need a corpus with phrase alignments annotated on sentence pairs. We extended the Syntactic Phrase Alignment Dataset for Evaluation (SPADE) (Arase and Tsujii, 2018), creating the Extended Syntactic Phrase Alignment DAtaset (ESPADA). Following the same annotation scheme, we annotated $1,916$ sentence pairs sampled from NIST OpenMT[4] corpora. ESPADA is now the largest annotation corpus for this problem and will be released by the Linguistic Data Consortium (LDC) soon.

A linguist first annotated gold-standard syntactic trees on paraphrases based on the head-driven phrase structure grammar. Then, three native or near-native English speakers annotated the $1,916$ paraphrases in parallel to identify phrasal paraphrases; *i.e.* the total number of annotated sentences is $5,748$. Before the formal annotation, there was a training phase to improve annotation agreement; all annotators annotated trial samples.[5] One of the authors inspected the results and gave advice on any misunderstandings of the annotation guidelines. Appendix C provides further details of the annotation process.

Table 1 shows the statistics for ESPADA and SPADE; $\sim$ 252k phrasal paraphrases were identified, among which $\sim$ 81k unique pairs were agreed upon by at least two annotators and $\sim$ 66k unique pairs were agreed upon by all annotators. The last two rows show, in ESPADA and SPADE, 3.2% to 4.7% of pairs did not satisfy the monotonicity condition, and 1.1% to 1.4% of triplets did not satisfy

[4] https://www.nist.gov/itl/iad/mig/openmt

[5] excluded from the formal annotation set

	ALIR (%)	ALIP (%)	ALIF (%)
ESPADA	93.3	90.2	91.7
SPADE (dev)	93.5	91.4	92.4
SPADE (test)	92.3	90.3	91.3

Table 2: Human performance

the familiness condition in alignments agreed upon by at least two annotators. Note that the monotonicity and familiness conditions are defined on relations of alignment pairs and triples, respectively; hence, these percentages do not mean that these percentages of alignments are non-compositional.

5.2 Evaluation Metrics and Upper Bounds

We used SPADE as an evaluation corpus; Table 1 shows statistics for its development (dev) and test sets. As evaluation metrics, we used alignment recall (ALIR), alignment precision (ALIP), and alignment F-measure (ALIF) (Arase and Tsujii, 2017, 2018). ALIR evaluates how gold-standard alignments can be replicated by automatic alignments, and ALIP measures how automatic alignments overlap with alignments identified by at least one annotator:

$$\text{ALIR} = \frac{|\{\mathbb{h} | \mathbb{h} \in \mathbb{H}_a \wedge \mathbb{h} \in \mathbb{G} \cap \mathbb{G}'\}|}{|\mathbb{G} \cap \mathbb{G}'|},$$

$$\text{ALIP} = \frac{|\{\mathbb{h} | \mathbb{h} \in \mathbb{H}_a \wedge \mathbb{h} \in \mathbb{G} \cup \mathbb{G}'\}|}{|\mathbb{H}_a|},$$

where $\mathbb{H}_a$ is a set of automatic alignments, and $\mathbb{G}$ and $\mathbb{G}'$ are those obtained by two respective annotators. ALIF computes the harmonic mean of ALIR and ALIP. Because SPADE provides alignments by three annotators, there are three combinations for $\mathbb{G}$ and $\mathbb{G}'$. The final ALIR, ALIP, and ALIF values are calculated by taking the averages.

Note that these evaluation metrics count null alignments also; hence, ALIP performs differently from a general precision metric in that stricter models will have lower ALIP scores. This is because a stricter model aligning only a small number of phrases ($\neq \tau_{\emptyset}$) increases the number of null alignments, making $|\mathbb{H}_a|$ larger.

The agreement among the human annotators can also be measured using ALIR, ALIP, and ALIF by regarding one annotator as a test and the other two as gold-standard and then taking averages. The scores for the trained annotators were consistent between ESPADA and SPADE as shown in Table 2. This indicates that phrase alignment is difficult even for humans because acceptable levels of semantic divergence in paraphrases cannot be perfectly controlled. Hence, we regard these human scores as upper bounds for ALIR, ALIP, and ALIF.

5.3 Comparison Method

As the comparison state-of-the-art syntactic phrase alignment method, we used Arase and Tsujii (2017). We re-implemented this method and compared the performance on aligning gold parse trees.

Additionally, we compared variations of our method via ablation studies. We investigated the effect of CTED by comparing it with alignments by a naive thresholding, which aligns phrases having cosine similarities above a threshold. The threshold was set to maximise the ALIF score on the SPADE development set.

To investigate the effect of our phrase representation model, we compared it with a simply fine-tuned BERT using Equation (1) but directly inputting basic phrase representations of $\boldsymbol{e}_p^t$ and $\boldsymbol{e}_n^t$. To investigate the effect of SimMatrix representation, we compared it with the representation of the [CLS] symbol (denoted as BERT+[CLS]). BERT defines its input to begin with the special symbol [CLS], whose representation has been commonly used as a representation of sentence pair (Devlin et al., 2019). The assumption here is that BERT may learn to embed information of similarity distribution into [CLS] representation.

As a pre-trained model for generating phrase representations, we compared the fine-tuning approach with the feature-based approach, *i.e.* FastText (Bojanowski et al., 2017) and embeddings from language models (ELMo) (Peters et al., 2018). For all pre-trained models, we used mean-pooling to generate a basic phrase representation, which consistently outperformed max-pooling in our preliminary experiments.

5.4 Model Settings

We used the following public pre-trained models: 'crawl-300d-2M-subword'[6] as FastText, 'Original (5.5B)'[7] as ELMo, and 'BERT-Base, Uncased'[8] as BERT. We implemented our method and its variations using PyTorch[9] with libraries Transformers,[10]

[6] https://dl.fbaipublicfiles.com/fasttext/vectors-english/crawl-300d-2M-subword.zip

[7] https://allennlp.org/elmo

[8] https://huggingface.co/bert-base-uncased

[9] https://pytorch.org/ (version 1.4.0)

[10] https://github.com/huggingface/transformers (version 0.6.2)

AllenNLP,[11] and NetworkX[12] for solving the minimum cost maximum flow problem in CTED.

Our attention mechanism had eight heads; the other settings were the same as those for Transformer (Vaswani et al., 2017). Dropouts of 10% and 50% were applied to the BERT and ELMo outputs, respectively, as recommended in their papers. The CNN had a kernel size of three in the convolution layer and two for the pooling layer. The SimMatrix was padded with zeros for sentences shorter than the maximum sequence length of 128.[13]

All models used AdamW (Loshchilov and Hutter, 2019) as an optimiser, using default settings except on the learning rate. We tuned a few hyperparameters in our models to maximise the ALIF score on the development set of SPADE by a grid search. The value of null alignment cost $\lambda_{\emptyset}$ was searched for in the range $[0.05, 0.95]$ by intervals of 0.05, the margin δ in the loss function was searched for in $[0.2, 1.0]$ by intervals of 0.2, and the learning rate was chosen from among $1.0\mathrm{e}-5$, $3.0\mathrm{e}-5$, and $5.0\mathrm{e}-5$.

5.5 Training Settings

All experiments were conducted on an NVIDIA Tesla V100 GPU. We trained our phrase representation model using ESPADA. We simply used all phrase alignments by the three annotators, regarding all of them as equally reliable, *i.e.* each sentence pair has three sets of phrase alignments. We split the entire dataset into training and validation sets (90% and 10%, respectively) after randomly shuffling the sentence pairs, which prevents the same sentence pair from appearing in both sets. The batch size was 16. Training was terminated by validation-based early-stopping with patience 5 and minimum delta 0.005.

To alleviate the randomness effects in initialising the neural networks, we trained and evaluated the models 10 times with random seeds and report means of the evaluation scores with 95% confidence intervals. Further, we tested the significance of differences in means of the evaluation scores by the randomised test (Efron and Tibshirani, 1994). Throughout the paper, we present the best scores with a significance level of $< 1\%$ using a bold font.

6 Experiment Results

6.1 Overall Results

Table 3 compares the methods' performance. BERT+SimMatrix+CTED (last row) includes the full feature set; it transforms the phrase representation using SimMatrix representation and aligns phrases using CTED. This method performed the best overall, achieving an ALIF score of 87.4% with post-processing. This ALIF score is 95.7% of that achieved by humans (Table 2).

We investigated non-compositional alignments produced by BERT+SimMatrix+CTED with post-processing. We found that 0.1% of alignment pairs did not satisfy the monotonicity condition and 1.2% of alignment triplets did not satisfy the familiness condition. These non-compositional alignments cover 3.5% and 23.2% of those of the gold standard that did not satisfy the monotonicity and familiness conditions, respectively (as shown in Table 1).

Effect of CTED Algorithm and Post-Processing The middle and last sets of rows compare CTED-based and thresholding-based alignments. Thresholding-based alignment greedily aligns phrases by disregarding compositionality. In contrast, the pure CTED-based alignment only allows compositional alignments and makes all non-compositional alignments null. Even though CTED is much stricter than thresholding, it achieved competitive ALIF scores. The scores of the CTED-based alignment further improves by allowing non-compositional alignments by post-processing; ALIR, ALIP, and ALIF improved by 2.2, 3.4, and 2.8 percentage points on average, respectively.

Effect of Phrase Representation Model The last set of rows shows the performance of alignments by CTED with different phrase representation approaches. BERT+SimMatrix+CTED significantly outperformed BERT+CTED and BERT+`[CLS]`+CTED. The superiority of SimMatrix representation over `[CLS]` was more pronounced on alignments with post-processing. Although ALIF of BERT+`[CLS]`+CTED with post-processing achieved 94.4% of the human score, SimMatrix representation further improved it by 1.2 percentage points.

These results indicate that a phrase representation that explicitly models the similarity distribution is crucial for handling non-compositional alignments. We conjecture that SimMatrix representation has two effects in phrase alignment.

[11] `https://allennlp.org/` (version 0.9.0)

[12] `https://networkx.github.io/` (version 2.4)

[13] None of the sentence pairs exceeded this limit.

		w/o post-processing			w/ post-processing		
Method	$\lambda_\emptyset$	ALIR (%)	ALIP (%)	ALIF (%)	ALIR (%)	ALIP (%)	ALIF (%)
Arase and Tsujii (2017)	–	81.8	75.6	78.6	–	–	–
BERT+Thresh.	0.80	83.6 ± 0.2	83.4 ± 0.2	83.5 ± 0.2	–	–	–
BERT+`[CLS]`+Thresh.	0.65	84.7 ± 0.5	83.5 ± 0.6	84.1 ± 0.5	–	–	–
BERT+SimMatrix+Thresh.	0.60	84.1 ± 0.4	84.7 ± 0.3	84.4 ± 0.3	–	–	–
BERT+CTED	0.90	85.3 ± 0.1	81.9 ± 0.1	83.5 ± 0.1	87.4 ± 0.2	85.2 ± 0.2	86.3 ± 0.2
BERT+`[CLS]`+CTED	0.80	85.6 ± 0.3	82.1 ± 0.5	83.8 ± 0.4	87.4 ± 0.4	85.0 ± 0.7	86.2 ± 0.5
BERT+SimMatrix+CTED	0.80	85.7 ± 0.2	82.7 ± 0.1	84.2 ± 0.2	**88.2** ± 0.3	**86.6** ± 0.2	**87.4** ± 0.2

Table 3: ALIR, ALIP, and ALIF scores with 95% confidence intervals

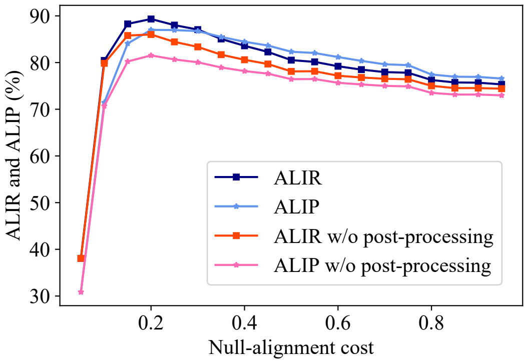

Figure 4: ALIR and ALIP on SPADE development set by null alignment cost

First, it encourages a null alignment in CTED when there is a more similar phrase beyond the local scope. *I.e.*, it implicitly relaxes the syntactic constraint when composing compositional alignments that could be too restrictive to handle non-compositional alignments. Second, the SimMatrix representation allows the post-processing to find a globally plausible alignment pair considering the entire similarity distribution.

Effect of Null-Alignment Cost Figure 4 presents ALIR and ALIP by the cost of null alignment $\lambda_\emptyset$ for BERT+SimMatrix+CTED with and without post-processing. A small $\lambda_\emptyset$ causes the method to align only a small number of phrases and produce a large number of null alignments. In contrast, a large $\lambda_\emptyset$ confuses the method by allowing a larger number of possible alignments. Both situations are harmful, but the former has a larger impact. This is because the constraint of CTED only allows a legitimate set of phrase alignments, which effectively prunes away incorrect alignments.

Figure 4 empirically confirms that the post-processing is effective in improving ALIR and ALIP scores; these scores with post-processing were always higher than those without. The same trend was also observed for BERT+CTED and for BERT+`[CLS]`+CTED. This occurs because our post-processing only allows non-compositional alignments of minimum cost. Hence, it also improves ALIR and ALIP scores when phrase representations are reliable.

6.2 Effects on Feature-Based Approaches

Table 4 shows the effect on performance when CTED is combined with the feature-based approaches: FastText, ELMo, and BERT without fine-tuning.[14] Specifically, we generated a phrase representation by simply mean-pooling token representations generated by these pre-trained models and aligned phrases by CTED or by thresholding. Note that these methods behave deterministically owing to the absence of neural network training.

BERT w/o fine-tuning+CTED achieved an ALIF score of 84.7% with post-processing, even though it only tunes the hyper-parameter $\lambda_\emptyset$. Although it scored lower than the proposed method (BERT+SimMatrix+CTED), the result is still encouraging for conducting phrase alignment in domains for which no corpora are available for training our phrase representation model.

Improvements in ALIR, ALIP, and ALIF scores by CTED over thresholding were much greater with FastText than with ELMo or BERT; it showed average gains of 6.0 to 8.6 percentage points. Improvements ranged from −0.8 to 2.7 for ELMo and from 0.2 to 1.3 percentage points for BERT. The CTED algorithm constrains alignments by the syntactic structures. FastText representations obviously do not retain such structural information. We conjecture that FastText-based alignment is com-

[14] Although we also applied our phrase representation model to feature-based approaches, the results were inferior to those given here, as discussed in Appendix D.

		w/o post-processing			w/ post-processing		
Method	$\lambda_\emptyset$	ALIR (%)	ALIP (%)	ALIF (%)	ALIR (%)	ALIP (%)	ALIF (%)
FastText+Thresh.	0.70	74.7	72.9	73.8	–	–	–
FastText+CTED	0.80	83.3	78.9	81.1	84.3	81.9	83.1
ELMo+Thresh.	0.50	81.7	80.7	81.2	–	–	–
ELMo+CTED	0.75	84.3	79.8	82.0	85.7	81.8	83.7
BERT w/o fine-tuning+Thresh.	0.80	82.3	79.6	80.9	–	–	–
BERT w/o fine-tuning+CTED	0.85	83.6	79.8	81.7	**85.9**	**83.5**	**84.7**

Table 4: ALIR, ALIP, and ALIF scores with feature-based approaches

pensated for by CTED. In contrast, the smaller improvements on ELMo and BERT imply that they obtain such structural information through their masked language model training. This result is consistent with previous studies (Hewitt and Manning, 2019; Jawahar et al., 2019; Reif et al., 2019) that confirmed that BERT learns syntactic structures.

7 Discussion and Future Work

In contrast to previous methods, ours can align phrases not only in paraphrasal sentence pairs but also in partially paraphrasal pairs. We plan to apply it to a comparable corpus of partial paraphrases and investigate the performance, with the aim of creating a large-scale syntactic and phrasal paraphrase dataset. We intend to expand our method to conduct forest alignments for making it robust against parsing errors, which are inevitable in handling large corpora. Further, as our method does not restrict input to syntactic trees but only assumes tree structures with arbitrary numbering (*e.g*. left-to-right post-order numbering) as input, we intend to try alignments of chunk-based trees, which is desirable for applications that process text fragments, *e.g*. those that perform information extraction.

Acknowledgments

Discussions on the tree mapping problem with Dr. Masakazu Ishihata and Mr. Taihei Oki have been illuminating. Dr. Joseph Tabolt, Mr. Michael Doerner, Ms. Tabatha Kobayashi, and Mr. Johannes Hoseth Klev made enormous contributions as annotators for creating ESPADA. We appreciate the anonymous reviewers for their insightful comments and suggestions to improve the paper. This work was supported by JST, ACT-I, Grant Number JPMJPR16U2, Japan.

References

Maytham Alabbas and Allan Ramsay. 2013. Optimising tree edit distance with subtrees for textual entailment. In *Proceedings of International Conference Recent Advances in Natural Language Processing*, pages 9–17.

Yuki Arase and Junichi Tsujii. 2017. Monolingual phrase alignment on parse forests. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–11.

Yuki Arase and Junichi Tsujii. 2018. SPADE: Evaluation dataset for monolingual phrase alignment. In *Proceedings of the Language Resources and Evaluation Conference (LREC).*

Yuki Arase and Jun'ichi Tsujii. 2019. Transfer fine-tuning: A BERT case study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5392–5403.

Marc Bernard, Laurent Boyer, Amaury Habrard, and Marc Sebban. 2008. Learning probabilistic models of tree edit distance. *Pattern Recognition*, 41(8):2611–2629.

Philip Bille. 2005. A survey on tree edit distance and related problems. *Theoretical Computer Science*, 337(1—3):217—239.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association of Computational Linguistics (TACL)*, 5:135–146.

David Burkett, John Blitzer, and Dan Klein. 2010. Joint parsing and alignment with weakly synchronized grammars. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 127–135.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1657–1668.

Do Kook Choe and David McClosky. 2015. Parsing paraphrases with joint inference. In *Proceedings of the Joint Conference of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 1223–1233.

Dipanjan Das and Noah A. Smith. 2009. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 468–476.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4171–4186.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 350–356.

Bradley Efron and R.J. Tibshirani. 1994. *An Introduction to the Bootstrap*. Chapman and Hall/CRC.

Hua He and Jimmy Lin. 2016. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 937–948.

Michael Heilman and Noah A. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 1011–1019.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4129–4138.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3651–3657.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A continuously growing dataset of sentential paraphrases. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1235–1245.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Pascual Martínez-Gómez and Yusuke Miyao. 2016. Rule extraction for tree-to-tree transducers by cost minimization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 12–22.

Martin McCaffery and Mark-Jan Nederhof. 2016. DTED: Evaluation of machine translation structure using dependency parsing and tree edit distance. In *Proceedings of the Workshop on Statistical Machine Translation (WMT)*, pages 491–498.

Yashar Mehdad. 2009. Automatic cost estimation for tree edit distance using particle swarm optimization. In *Proceedings of the Joint Conference of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 289–292.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2249–2255.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 2227–2237.

Vasin Punyakanok, Dan Roth, and Wen tau Yih. 2004. Mapping dependencies trees: An application to question answering. International Symposium on Artificial Intelligence and Mathematics (Special Session).

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B Viegas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of BERT. In *Proceedings of Conference on Neural Information Processing Systems (NeurIPS)*, pages 8594–8603.

David Smith and Jason Eisner. 2006. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *Proceedings of the Workshop on Statistical Machine Translation (WMT)*, pages 23–30.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of Conference on Neural Information Processing Systems (NeurIPS)*, pages 5998–6008.

Mengqiu Wang and Christopher Manning. 2010. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 1164–1172.

Jonathan Weese, Juri Ganitkevitch, and Chris Callison-Burch. 2014. PARADIGM: Paraphrase diagnostics through grammar matching. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 192–201.

Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. 2013. Answer extraction as sequence tagging with tree edit distance. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 858–867.

Da Yin, Tao Meng, and Kai-Wei Chang. 2020. SentiBERT: A transferable transformer-based architecture for compositional sentiment semantics. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 3695–3706.

Kaizhong Zhang. 1996. A constrained edit distance between unordered labeled trees. *Algorithmica*, 15(3):205–222.

Kaizhong Zhang and Tao Jiang. 1994. Some MAX SNP-hard results concerning unordered labeled trees. *Information Processing Letters*, 49(5):249–254.

Kaizhong Zhang, Jason T. L. Wang, and Dennis Shasha. 1995. On the editing distance between undirected acyclic graphs and related problems. In *Annual Symposium on Combinatorial Pattern Matching*, pages 395–407.

Appendices

A Detailed Comparison with Previous Study

Arase and Tsujii (2017) include additional conditions that a legitimate set of compositional alignments should satisfy. One of these is called the root-pair containment condition, which requires the root nodes of trees to be aligned. This constraint firmly restricts their method such that it can only handle a paraphrasal sentence pair as input. Our method, by contrast, can align any pair of sentences, *i.e.* not only paraphrasal sentences but also sentences that are only partially paraphrasal.

Algorithm B.1 CTED algorithm (Zhang, 1996)

```
Input: Source and target trees $T^s$ and $T^t$
Output: Tree edit distance matrix $D[i][j]$, where
    $1 \leq i \leq |T^s|$ and $1 \leq j \leq |T^t|$
$D[0][0] = 0, F[0][0] = 0$
for all $i = 1$ to $|T^s|$ do                ▷ target side is $\tau_\emptyset$
    $F[i][0] = \sum_{k=1}^{n_i} D[i_k][0]$
    $D[i][0] = F[i][0] + \gamma(\langle \tau_i^s, \tau_\emptyset \rangle)$
for all $j = 1$ to $|T^t|$ do                ▷ source side is $\tau_\emptyset$
    $F[0][j] = \sum_{k=1}^{n_j} D[0][j_k]$
    $D[0][j] = F[0][j] + \gamma(\langle \tau_\emptyset, \tau_j^t \rangle)$
for all $i = 1$ to $|T^s|$ do
    for all $j = 1$ to $|T^t|$ do
        Compute $F[i][j]$ (Equation (2))
        Compute $D[i][j]$ (Equation (3))
```

Additionally, in their study, the familiness condition is replaced by the maximum set condition. The maximum set condition, together with the monotonicity condition, constrains all the lowest common ancestors (LCAs) of any pair of non-null alignments in $\mathbb{H}_L$ to ensure that they are aligned. That is, for all $\mathbb{h}_m, \mathbb{h}_n \in \mathbb{H}_L$ of non-null alignments, $\langle \tau_i^s, \tau_i^t \rangle \in \mathbb{H}_L$, where $\tau_i^s = \text{lca}(\tau_m^s, \tau_n^s)$ and $\tau_i^t = \text{lca}(\tau_m^t, \tau_n^t)$. Owing to this constraint, their method belongs to the class of LCA-preserving distance mappings (Zhang et al., 1995), whose constraint is tighter than the constraint edit distance mapping. In phrase alignment, this forces LCAs of two aligned nodes to be aligned as well, even though the majority of phrases under the LCAs are null alignments. By contrast, CTED allows such LCAs to have null alignments depending on the alignments of descendant nodes.

B CTED Algorithm

Algorithm B.1 shows the CTED algorithm. For brevity, we denote the ith node in a tree as i and its child nodes as $I = \{i_k | i_1, \ldots, i_{n_i}\}$, where n_i is the number of children. The input trees are numbered; the numbers are determined by an arbitrary ordering of the nodes in the tree, such as left-to-right post-order numbering or left-to-right pre-order numbering. The algorithm first computes the minimum cost of $\mathbb{H}_{i,j}$, which are alignments of forests rooted at nodes i and j (line 10):

$$F[i][j] = \min \begin{cases} F[0][j] + \min_{1 \le k \le n_j}\{F[i][j_k] - F[0][j_k]\}, \\ F[i][0] + \min_{1 \le l \le n_i}\{F[i_l][j] - F[i_l][0]\}, \\ \min_{\mathbb{H}_{i,j}} \gamma(\mathbb{H}_{i,j}). \end{cases} \tag{2}$$

Then (line 11), it computes the minimum cost of $\langle i, j\rangle$, $\langle i, \tau_\emptyset\rangle$, and $\langle \tau_\emptyset, j\rangle$ as

$$D[i][j] = \min \begin{cases} D[0][j] + \min_{1 \le k \le n_j}\{D[i][j_k] - D[0][j_k]\}, \\ D[i][0] + \min_{1 \le l \le n_i}\{D[i_l][j] - D[i_l][0]\}, \\ F[i][j] + \gamma(\langle \tau_i^s, \tau_j^t\rangle). \end{cases} \tag{3}$$

In Equation (2), $\gamma(\mathbb{H}_{i,j})$ is the summation of the alignment costs between the forests:

$$\gamma(\mathbb{H}_{i,j}) = \sum_{\langle u,v\rangle \in \mathbb{H}_{i,j}} \gamma(\langle u, v\rangle),$$

where u or v is $\tau_\emptyset$ for null alignments.

The algorithm searches for $\mathbb{H}_{i,j}$ that has the minimum cost by solving the minimum cost maximum flow problem on a graph $G(V, E)$, as shown in Figure 5. The vertex set consists of $V = \{s_0, s_t, \tau_\emptyset^i, \tau_\emptyset^j\} \cup I \cup J$, where s_0 and s_t are the start and sink nodes, respectively, and $\tau_\emptyset^i$ and $\tau_\emptyset^j$ are null nodes. Each edge in E has a cost and capacity: Edges (s_0, i_k), $(s_0, \tau_\emptyset^i)$, (j_l, s_t), and $(\tau_\emptyset^j, s_t)$ are cost zero; (i_k, j_l) is cost $D[i_k][j_l]$; $(\tau_\emptyset^i, j_t)$ is cost $D[0][j_l]$; $(i_k, \tau_\emptyset^j)$ is cost $D[i_k][0]$; and $(\tau_\emptyset^i, \tau_\emptyset^j)$ is cost zero. All the edges have capacity one except $(s_0, \tau_\emptyset^i)$, $(\tau_\emptyset^i, \tau_\emptyset^j)$, and $(\tau_\emptyset^j, s_t)$, whose capacities are n_j, $\min(n_i, n_j)$,[15] and n_i, respectively. Obviously, the maximum flow of G is $n_i + n_j$ and G is a network with integer capacities and non-negative costs. The minimum cost on the maximum flow of G is proven to be in agreement with $\min_{\mathbb{H}_{i,j}} \gamma(\mathbb{H}_{i,j})$ in (Zhang, 1996).

Algorithm B.1 only shows the computation of the alignment cost for brevity. However, the corresponding edit operations, *i.e.* alignments, can be computed simultaneously in the same manner as the edit cost.

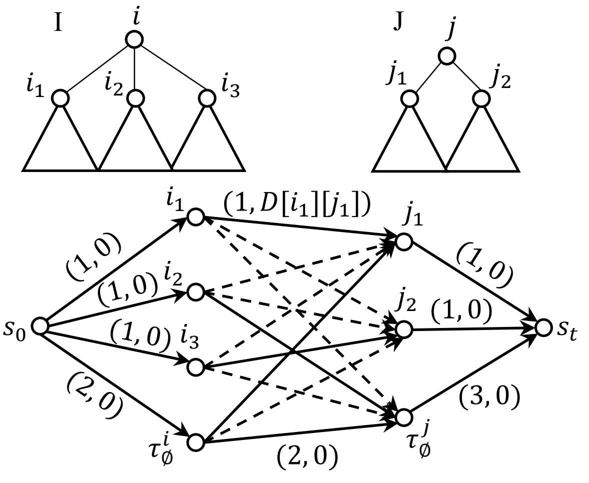

Figure 5: Minimum cost maximum flow problem (values in parentheses represent the (capacity, cost) of each edge)

C Details of ESPADA Creation

To obtain paraphrasal sentence pairs to annotate, we sampled paraphrases from reference translations in NIST OpenMT corpora[16] excluding sentences in SPADE. There are a variety of resources for constructing paraphrases, including reference translations (Weese et al., 2014), news texts (Dolan et al., 2004), and tweets (Lan et al., 2017). Arase and Tsujii (2018) discussed how paraphrases constructed from reference translations are authentic in the sense that they only pose paraphrastic phenomena because they are constrained by corresponding source sentences. By contrast, paraphrases extracted from other resources tend to have more diverse linguistic phenomena, such as additions and omissions of information and inferences requiring knowledge of the world.

First, we recruited a linguist who is also a native English speaker to annotate the gold-standard syntactic trees on paraphrases based on the grammar of the head-driven phrase structure. Through this process, the linguist identified and discarded ungrammatical and/or non-paraphrasal pairs. The annotated trees were checked automatically for formatting, and the linguist corrected the annotations of trees with errors, such as trees with inconsistent bracketing. We then had three native or near-native English speakers annotate the phrase alignments.

[15] The flows in a solution should pass through all the non-null nodes; hence, the capacity between empty trees should be $(n_i + n_j) - \max(n_i, n_j) = \min(n_i, n_j)$, which subtracts the minimum flows from/to non-null nodes from the maximum flow of G. Zhang (1996) set this capacity to $\max(n_i, n_j) - \min(n_i, n_j)$, but that produces a degenerate solution.

		w/o post-processing			w/ post-processing		
Method	$\lambda_\emptyset$	ALIR (%)	ALIP (%)	ALIF (%)	ALIR (%)	ALIP (%)	ALIF (%)
ELMo+SimMatrix+CTED	0.60	82.5 ± 0.1	79.9 ± 0.1	81.2 ± 0.1	83.5 ± 0.1	81.6 ± 0.1	82.5 ± 0.1
BERT w/o FT+ `[CLS]` +CTED	0.75	83.8 ± 0.1	80.3 ± 0.1	82.0 ± 0.1	85.5 ± 0.2	82.9 ± 0.2	84.2 ± 0.2
BERT w/o FT+SimMatrix+CTED	0.80	84.3 ± 0.1	80.4 ± 0.0	82.3 ± 0.0	85.3 ± 0.1	82.0 ± 0.1	83.6 ± 0.1

Table 5: ALIR, ALIP, and ALIF scores for our phrase representation model when applied to feature-based models (‘BERT w/o FT’ stands for ‘BERT without fine-tuning’)

D Phrase Representation with Feature-Based Approaches

We also applied our phrase representation model to ELMo and BERT, using them as feature generators. We trained only the attention and CNN models using ESPADA. For ELMo, we also trained the scalar weighting parameters.

Table 5 shows the results. Unfortunately, all of these methods are inferior to their counterparts that lack our phrase representation model: ELMo+CTED and BERT w/o fine-tuning+CTED, respectively. We conjecture that ESPADA may be insufficiently large for training our phrase representation model to adapt to a pre-trained model that behaves in a completely independent manner. BERT’s ability to adapt quickly to a specific task by fine-tuning is a notable advantage.

[16] LDC catalogue numbers: LDC2010T14, LDC2010T17, LDC2010T21, LDC2010T23, LDC2013T03