Back to Basics for Monolingual Alignment: Exploiting Word Similarity and Contextual Evidence

Md Arafat Sultan[†], Steven Bethard[‡] and Tamara Sumner[†]
[†]Institute of Cognitive Science and Department of Computer Science
University of Colorado Boulder
[‡]Department of Computer and Information Sciences
University of Alabama at Birmingham
arafat.sultan@colorado.edu, bethard@cis.uab.edu, sumner@colorado.edu

Abstract

We present a simple, easy-to-replicate monolingual aligner that demonstrates state-of-the-art performance while relying on almost no supervision and a very small number of external resources. Based on the hypothesis that words with similar meanings represent potential pairs for alignment if located in similar contexts, we propose a system that operates by finding such pairs. In two intrinsic evaluations on alignment test data, our system achieves F_1 scores of 88–92%, demonstrating 1–3% absolute improvement over the previous best system. Moreover, in two extrinsic evaluations our aligner outperforms existing aligners, and even a naive application of the aligner approaches state-of-the-art performance in each extrinsic task.

1 Introduction

Monolingual alignment is the task of discovering and aligning similar semantic units in a pair of sentences expressed in a natural language. Such alignments provide valuable information regarding how and to what extent the two sentences are related. Consequently, alignment is a central component of a number of important tasks involving text comparison: textual entailment recognition, textual similarity identification, paraphrase detection, question answering and text summarization, to name a few.

The high utility of monolingual alignment has spawned significant research on the topic in the recent past. Major efforts that have treated alignment as a standalone problem (MacCartney et al., 2008; Thadani and McKeown, 2011; Yao et al., 2013a) are primarily supervised, thanks to the manually aligned corpus with training and test sets from Microsoft Research (Brockett, 2007). Primary concerns of such work include both quality and speed, due to the fact that alignment is frequently a component of larger NLP tasks.

Driven by similar motivations, we seek to devise a lightweight, easy-to-construct aligner that produces high-quality output and is applicable to various end tasks. Amid a variety of problem formulations and ingenious approaches to alignment, we take a step back and examine closely the effectiveness of two frequently made assumptions: 1) Related semantic units in two sentences must be similar or related in their meaning, and 2) Commonalities in their semantic contexts in the respective sentences provide additional evidence of their relatedness (MacCartney et al., 2008; Thadani and McKeown, 2011; Yao et al., 2013a; Yao et al., 2013b). Alignment, based solely on these two assumptions, reduces to finding the best combination of pairs of similar semantic units in similar contexts.

Exploiting existing resources to identify similarity of semantic units, we search for robust techniques to identify contextual commonalities. Dependency trees are a commonly used structure for this purpose. While they remain a central part of our aligner, we expand the horizons of dependency-based alignment beyond exact matching by systematically exploiting the notion of "type equivalence" with a small handcrafted set of equivalent dependency types. In addition, we augment dependency-based alignment with surface-level text analysis.

While phrasal alignments are important and have

Transactions of the Association for Computational Linguistics, 2 (2014) 219–230. Action Editor: Alexander Koller.
Submitted 11/2013; Revised 1/2014; Published 5/2014.

been investigated in multiple studies, we focus primarily on word alignments (which have been shown to form the vast majority of alignments ($\geq$ 95%) in multiple human-annotated corpora (Yao et al., 2013b)), keeping the framework flexible enough to allow incorporation of phrasal alignments in future.

Evaluation of our aligner on the benchmark dataset reported in (Brockett, 2007) shows an F_1 score of 91.7%: a 3.1% absolute improvement over the previous best system (Yao et al., 2013a), corresponding to a 27.2% error reduction. It shows superior performance also on the dataset reported in (Thadani et al., 2012). Additionally, we present results of two extrinsic evaluations, namely textual similarity identification and paraphrase detection. Our aligner not only outperforms existing aligners in each task, but also approaches top systems for the extrinsic tasks.

2 Background

Monolingual alignment has been applied to various NLP tasks including textual entailment recognition (Hickl et al., 2006; Hickl and Bensley, 2007), paraphrase identification (Das and Smith, 2009; Madnani et al., 2012), and textual similarity assessment (Bär et al., 2012; Han et al., 2013) – in some cases explicitly, i.e., as a separate module. But many such systems resort to simplistic and/or ad-hoc strategies for alignment and in most such work, the alignment modules were not separately evaluated on alignment benchmarks, making their direct assessment difficult.

With the introduction of the MSR alignment corpus (Brockett, 2007) developed from the second Recognizing Textual Entailment challenge data (Bar-Haim et al., 2006), direct evaluation and comparison of aligners became possible. The first aligner trained and evaluated on the corpus was a phrasal aligner called MANLI (MacCartney et al., 2008). It represents alignments as sets of different *edit* operations (where a sequence of edits turns one input sentence into the other) and finds an optimal set of edits via a simulated annealing search. Weights of different edit features are learned from the training set of the MSR alignment corpus using a perceptron learning algorithm. MANLI incorporates only shallow features characterizing contextual similarity: relative positions of the two phrases being aligned (or not) in the two sentences and boolean features representing whether or not the preceding and following tokens of the two phrases are similar.

Thadani and McKeown (2011) substituted MANLI's simulated annealing-based decoding with integer linear programming, and achieved a considerable speed-up. More importantly for our discussion, they found contextual evidence in the form of syntactic constraints useful in better aligning stop words. Thadani et al. (2012) further extended the model by adding features characterizing dependency *arc edits*, effectively bringing stronger influence of contextual similarity into alignment decisions. Again the performance improved consequently.

The most successful aligner to date both in terms of accuracy and speed, called JacanaAlign, was developed by Yao et al. (2013a). In contrast to the earlier systems, JacanaAlign is a word aligner that formulates alignment as a sequence labeling problem. Each word in the source sentence is labeled with the corresponding target word index if an alignment is found. It employs a conditional random field to assign the labels and uses a feature set similar to MANLI's in terms of the information they encode (with some extensions). Contextual features include only semantic match of the left and the right neighbors of the two words and their POS tags. Even though JacanaAlign outperformed the MANLI enhancements despite having less contextual features, it is difficult to compare the role of context in the two models because of the large paradigmatic disparity. An extension of JacanaAlign was proposed for phrasal alignments in (Yao et al., 2013b), but the contextual features remained largely unchanged.

Noticeable in all the above systems is the use of contextual evidence as a feature for alignment, but in our opinion, not to an extent sufficient to harness its full potential. Even though deeper dependency-based modeling of contextual commonalities can be found in some other studies (Kouylekov and Magnini, 2005; Chambers et al., 2007; Chang et al., 2010; Yao et al., 2013c), we believe there is further scope for systematic exploitation of contextual evidence for alignment, which we aim to do in this work.

On the contrary, word semantic similarity has been a central component of most aligners; various measures of word similarity have been utilized, including string similarity, resource-based similarity (derived from one or more lexical resources like WordNet)

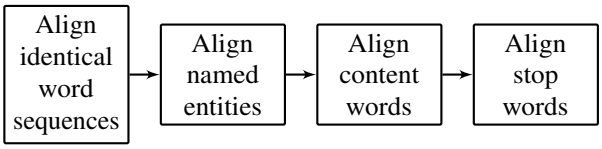


Figure 1: System overview

and distributional similarity (computed from word co-occurrence statistics in large corpora). An important trade-off between precision, coverage and speed exists here and aligners commonly rely on only a subset of these measures (Thadani and McKeown, 2011; Yao et al., 2013a). We use the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013), which is a large resource of lexical and phrasal paraphrases constructed using bilingual pivoting (Bannard and Callison-Burch, 2005) over large parallel corpora.

3 System

Our system operates as a pipeline of alignment modules that differ in the types of word pairs they align. Figure 1 is a simplistic representation of the pipeline. Each module makes use of contextual evidence to make alignment decisions. In addition, the last two modules are informed by a word semantic similarity algorithm. Because of their phrasal nature, we treat named entities separately from other content words. The rationale behind the order in which the modules are arranged is discussed later in this section (3.3.5).

Before discussing each alignment module in detail, we describe the system components that identify word and contextual similarity.

3.1 Word Similarity

The ability to correctly identify semantic similarity between words is crucial to our aligner, since contextual evidence is important only for similar words. Instead of treating word similarity as a continuous variable, we define three levels of similarity.

The first level is an exact word or lemma match which is represented by a similarity score of 1. The second level represents semantic similarity between two terms which are not identical. To identify such word pairs, we employ the Paraphrase Database (PPDB)[1]. We use the largest (XXXL) of the PPDB's lexical paraphrase packages and treat all pairs identically by ignoring the accompanying statistics. We customize the resource by removing pairs of identical words or lemmas and adding lemmatized forms of the remaining pairs. For now, we use the term *ppdbSim* to refer to the similarity of each word pair in this modified version of PPDB (which is a value in (0, 1)) and later explain how we determine it (Section 3.3.5). Finally, any pair of different words which is absent in PPDB is assigned a zero similarity score.

[1] http://paraphrase.org

3.2 Extracting Contextual Evidence

Our alignment modules collect contextual evidence from two complementary sources: syntactic dependencies and words occurring within a small textual vicinity of the two words to be aligned. The application of each kind assumes a common principle of *minimal* evidence. Formally, given two input sentences S and T, we consider two words $s \in S$ and $t \in T$ to form a candidate pair for alignment if $\exists r_s \in S$ and $\exists r_t \in T$ such that:

1. $(s,t) \in \Re_{Sim}$ and $(r_s, r_t) \in \Re_{Sim}$, where $\Re_{Sim}$ is a binary relation indicating *sufficient* semantic relatedness between the members of each pair ($\geq ppdbSim$ in our case).
2. $(s, r_s) \in \Re_{C_1}$ and $(t, r_t) \in \Re_{C_2}$, such that $\Re_{C_1} \approx \Re_{C_2}$; where $\Re_{C_1}$ and $\Re_{C_2}$ are binary relations representing specific types of contextual relationships between two words in a sentence (e.g., an *nsubj* dependency between a verb and a noun). The symbol $\approx$ represents equivalence between two relationships, including identicality.

Note that the minimal-evidence assumption holds a single piece of contextual evidence as sufficient support for a potential alignment; but as we discuss later in this section, an evidence for word pair (s,t) (where $s \in S$ and $t \in S$) may not lead to an alignment if there exists a competing pair (s',t) or (s,t') with more evidence (where $s' \in S$ and $t' \in T$).

In the rest of this section, we elaborate the different forms of contextual relationships we exploit along with the notion of equivalence between relationships.

3.2.1 Syntactic Dependencies

Dependencies can be important sources of contextual evidence. Two *nsubj* children r_s and r_t of two verbs $s \in S$ and $t \in T$, for example, provide evidence for not only an (s,t) alignment, but

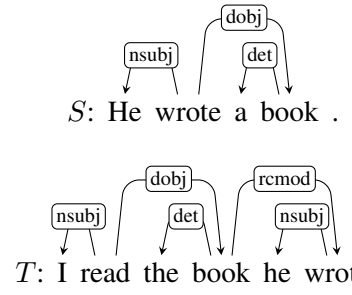


Figure 2: Equivalent dependency types: *dobj* and *rcmod*

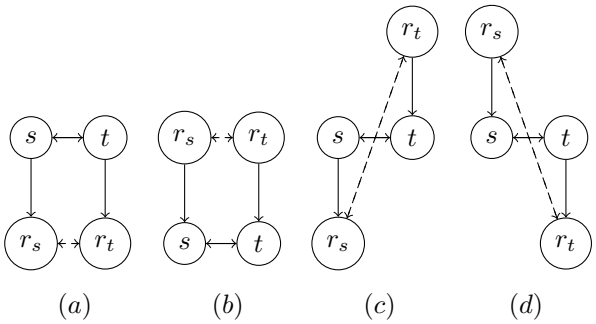


Figure 3: Parent-child orientations in dependencies

also an (r_s, r_t) alignment if $(s, t) \in \Re_{Sim}$ and $(r_s, r_t) \in \Re_{Sim}$. (We adopt the Stanford typed dependencies (de Marneffe and Manning, 2008).)

Moreover, dependency types can exhibit equivalence; consider the two sentences in Figure 2. The *dobj* dependency in S is equivalent to the *rcmod* dependency in T (*dobj* $\approx$ *rcmod*, following our earlier notation) since they represent the same semantic relation between identical word pairs in the two sentences. To be able to use such evidence for alignment, we need to go beyond exact matching of dependencies and develop a mapping among dependency types that encodes such equivalence. Note also that the parent-child roles are opposite for the two dependency types in the above example, a scenario that such a mapping must accommodate.

The four possible such scenarios regarding parent-child orientations are shown in Figure 3. If $(s, t) \in \Re_{Sim}$ and $(r_s, r_t) \in \Re_{Sim}$ (represented by bidirectional arrows), then each orientation represents a set of possible ways in which the S and T dependencies (unidirectional arrows) can provide evidence of similarity between the contexts of s in S and t in T. Each such set comprises equivalent dependency type pairs for that orientation. In the example of Figure 2, *(dobj, rcmod)* is such a pair for orientation (c), given $s = t =$ "*wrote*" and $r_s = r_t =$ "*book*".

We apply the notion of dependency type equivalence to intra-category alignment of content words in four major lexical categories: *verbs*, *nouns*, *adjectives* and *adverbs* (the Stanford POS tagger (Toutanova et al., 2003) is used to identify the categories). Table 1 shows dependency type equivalences for each lexical category of s and t.

The '$\leftarrow$' sign on column 5 of some rows represents a duplication of the column 4 content of the same row. For each row, columns 4 and 5 show two sets of dependency types; each member of the first is equivalent to each member of the second for the current orientation (column 1) and lexical categories of the associated words (columns 2 and 3). For example, row 2 represents the fact that an *agent* relation (between s and r_s; s is the parent) is equivalent to an *nsubj* relation (between t and r_t; t is the parent).

Note that the equivalences are fundamentally redundant across different orientations. For example, row 2 (which is presented as an instance of orientation *(a)*) can also be presented as an instance of orientation *(b)* with *POS*(s)=*POS*(t)=*noun* and *POS*(r_s)=*POS*(r_t)=*verb*. We avoid such redundancy for compactness. For the same reason, the equivalence of *dobj* and *rcmod* in Figure 2 is shown in the table only as an instance of orientation *(c)* and not as an instance of orientation *(d)* (in general, this is why orientations *(b)* and *(d)* are absent in the table).

We present dependency-based contextual evidence extraction in Algorithm 1. (The Stanford dependency parser (de Marneffe et al., 2006) is used to extract the dependencies.) Given a word pair (s_i, t_j) from the input sentences S and T, it collects contextual evidence (as indexes of r_{s_i} and r_{t_j} with a positive similarity) for each matching row in Table 1. An exact match of the two dependencies is also considered a piece of evidence. Note that Table 1 only considers content word pairs (s_i, t_j) such that *POS*(s_i)=*POS*(t_j), but as 90% of all content word alignments in the MSR alignment *dev* set are within the same lexical category, this is a reasonable set to start with.

3.2.2 Textual Neighborhood

While equivalent dependencies can provide strong contextual evidence, they can not ensure high recall because, a) the ability to accurately extract depen-

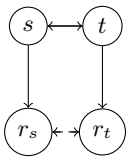
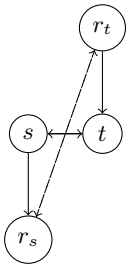
Orientation	POS(s,t)	POS(r_s,r_t)	S Dependency Types	T Dependency Types
 (a)	verb	verb	{purpcl, xcomp}	⟵
		noun	{agent, nsubj, xsubj}	⟵
			{dobj, nsubjpass, rel}	⟵
			{tmod, prep_in, prep_at, prep_on}	⟵
			{iobj, prep_to}	⟵
	noun	verb	{infmod, partmod, rcmod}	⟵
		noun	{pos, nn, prep_of, prep_in, prep_at, prep_for}	⟵
		adjective	{amod, rcmod}	⟵
 (c)	verb	verb	{conj_and}	⟵
			{conj_or}	⟵
			{conj_nor}	⟵
		noun	{dobj, nsubjpass, rel}	{infmod, partmod, rcmod}
		adjective	{acomp}	{cop, csubj}
	noun	noun	{conj_and}	⟵
			{conj_or}	⟵
			{conj_nor}	⟵
		adjective	{amod, rcmod}	{nsubj}
	adjective	adjective	{conj_and}	⟵
			{conj_or}	⟵
			{conj_nor}	⟵
	adverb	adverb	{conj_and}	⟵
			{conj_or}	⟵
			{conj_nor}	⟵

Table 1: Equivalent dependency structures

Algorithm 1: $depContext(S, T, i, j, EQ)$

Input:
1. S, T: Sentences to be aligned
2. i: Index of a word in S
3. j: Index of a word in T
4. EQ: Dependency type equivalences (Table 1)

Output: $context$ = $\{(k, l)\}$: pairs of word indexes

$context \leftarrow \{(k,l) : wordSim(s_k, t_l) > 0$
$\quad \wedge (i, k, \tau_s) \in dependencies(S)$
$\quad \wedge (j, l, \tau_t) \in dependencies(T)$
$\quad \wedge POS(s_i) = POS(t_j) \wedge POS(s_k) = POS(t_l)$
$\quad \wedge (\tau_s = \tau_t$
$\qquad \vee (POS(s_i), POS(s_k), \tau_s, \tau_t) \in EQ))\}$

Algorithm 2: $textContext(S, T, i, j, \text{STOP})$

Input:
1. S, T: Sentences to be aligned
2. i: Index of a word in S
3. j: Index of a word in T
4. STOP: A set of stop words

Output: $context$ = $\{(k, l)\}$: pairs of word indexes

$C_i \leftarrow \{k : k \in [i-3, i+3] \wedge k \neq i \wedge s_k \notin \text{STOP}\}$
$C_j \leftarrow \{l : l \in [j-3, j+3] \wedge l \neq j \wedge t_l \notin \text{STOP}\}$
$context \leftarrow C_i \times C_j$

dencies is limited by the accuracy of the parser, and b) we investigate equivalence types for only inter-lexical-category alignment in this study. Therefore we apply an additional model of word context: the textual neighborhood of s in S and t in T.

Extraction of contextual evidence for content words from textual neighborhood is described in Algorithm 2. Like the dependency-based module, it accumulates evidence for each (s_i, t_j) pair by inspecting multiple pairs of neighboring words. But instead of aligning only words within a lexical category, this module also performs inter-category alignment, considering content words within a $(3, 3)$ window of s_i and t_j as neighbors. We implement relational equivalence ($\approx$) here by holding any two positions within the window equally contributive and mutually comparable as sources of contextual evidence.

3.3 The Alignment Algorithm

We now describe each alignment module in the pipeline and their sequence of operation.

3.3.1 Identical Word Sequences

The presence of a common word sequence in S and T is indicative of an (a) identical, and (b) con-

textually similar word in the other sentence for each word in the sequence. We observe the results of aligning identical words in such sequences of length n containing at least one content word. This simple heuristic demonstrates a high precision ($\approx 97\%$) on the MSR alignment *dev* set for $n \geq 2$, and therefore we consider membership in such sequences as the simplest form of contextual evidence in our system and align all identical word sequence pairs in S and T containing at least one content word. From this point on, we will refer to this module as *wsAlign*.

3.3.2 Named Entities

We align named entities separately to enable the alignment of full and partial mentions (and acronyms) of the same entity. We use the Stanford Named Entity Recognizer (Finkel et al., 2005) to identify named entities in S and T. After aligning the exact term matches, any unmatched term of a partial mention is aligned to all terms in the full mention. The module recognizes only first-letter acronyms and aligns an acronym to all terms in the full mention of the corresponding name.

Since named entities are instances of nouns, named entity alignment is also informed by contextual evidence (which we discuss in the next section), but happens before alignment of other generic content words. Parents (or children) of a named entity are simply the parents (or children) of its head word. We will refer to this module as a method named *neAlign* from this point on.

3.3.3 Content Words

Extraction of contextual evidence for promising content word pairs has already been discussed in Section 3.2, covering both dependency-based context and textual context.

Algorithm 3 (*cwDepAlign*) describes the dependency-based alignment process. For each potentially alignable pair (s_i, t_j), the dependency-based context is extracted as described in Algorithm 1, and context similarity is calculated as the sum of the word similarities of the (s_k, t_l) context word pairs (lines 2-7). (The *wordSim* method returns a similarity score in {*0, ppdbSim, 1*}.) The alignment score of the (s_i, t_j) pair is then a weighted sum of word and contextual similarity (lines 8-12). (We discuss how the weights are set in Section 3.3.5.) The module then aligns (s_i, t_j) pairs with non-zero evidence in decreasing order of this score (lines 13-18). In addition, it aligns all the pairs that contributed contextual evidence for the (s_i, t_j) alignment (lines 19-22). Note that we implement a one-to-one alignment whereby a word gets aligned at most once within the module.

Algorithm 3: $cwDepAlign(S, T, EQ, A_E, w, \text{STOP})$

Input:
1. S, T: Sentences to be aligned
2. EQ: Dependency type equivalences (Table 1)
3. A_E: Already aligned word pair indexes
4. w: Weight of word similarity relative to contextual similarity
5. STOP: A set of stop words

Output: $A = \{(i, j)\}$: word index pairs of aligned words $\{(s_i, t_j)\}$ where $s_i \in S$ and $t_j \in T$

1 $\Psi \leftarrow \varnothing$; $\Lambda_\Psi \leftarrow \varnothing$; $\Phi \leftarrow \varnothing$
2 **for** $s_i \in S, t_j \in T$ **do**
3 **if** $s_i \notin \text{STOP} \wedge \neg\exists t_l : (i, l) \in A_E$
4 $\wedge\, t_j \notin \text{STOP} \wedge \neg\exists s_k : (k, j) \in A_E$
5 $\wedge\, wordSim(s_i, t_j) > 0$ **then**
6 *context* $\leftarrow depContext(S, T, i, j, EQ)$
   $contextSim \leftarrow \sum_{(k,l) \in context} wordSim(s_k, t_l)$
7
8 **if** $contextSim > 0$ **then**
9 $\Psi \leftarrow \Psi \cup \{(i, j)\}$
10 $\Lambda_\Psi(i, j) \leftarrow$ *context*
11 $\Phi(i, j) \leftarrow w * wordSim(s_i, t_j)$
12 $+(1 - w) * contextSim$
13 Linearize and sort Ψ in decreasing order of $\Phi(i, j)$
14 $A \leftarrow \varnothing$
15 **for** $(i, j) \in \Psi$ **do**
16 **if** $\neg\exists l : (i, l) \in A$
17 $\wedge\neg\exists k : (k, j) \in A$ **then**
18 $A \leftarrow A \cup \{(i, j)\}$
19 **for** $(k, l) \in \Lambda_\Psi(i, j)$ **do**
20 **if** $\neg\exists q : (k, q) \in A \cup A_E$
21 $\wedge\neg\exists p : (p, l) \in A \cup A_E$ **then**
22 $A \leftarrow A \cup \{(k, l)\}$

Algorithm 4 (*cwTextAlign*) presents alignment based on similarities in the textual neighborhood. For each potentially alignable pair (s_i, t_j), Algorithm 2 is used to extract the context, which is a set of neighboring content word pairs (lines 2-7). The contextual similarity is the sum of the similarities of these pairs

Algorithm 4: $cwTextAlign(S, T, A_E, w, \text{STOP})$

Input:
1. S, T: Sentences to be aligned
2. A_E: Existing alignments by word indexes
3. w: Weight of word similarity relative to contextual similarity
4. STOP: A set of stop words

Output: $A = \{(i, j)\}$: word index pairs of aligned words $\{(s_i, t_j)\}$ where $s_i \in S$ and $t_j \in T$

1 $\Psi \leftarrow \varnothing$; $\Phi \leftarrow \varnothing$
2 **for** $s_i \in S$, $t_j \in T$ **do**
3 **if** $s_i \notin \text{STOP} \wedge \neg\exists t_l : (i, l) \in A_E$
4 $\wedge\, t_j \notin \text{STOP} \wedge \neg\exists s_k : (k, j) \in A_E$
5 $\wedge\, wordSim(s_i, t_j) > 0$ **then**
6 $\Psi \leftarrow \Psi \cup \{(i, j)\}$
7 $context \leftarrow textContext(S, T, i, j, \text{STOP})$
 $contextSim \leftarrow \sum_{(k,l) \in context} wordSim(s_k, t_l)$
8
9 $\Phi(i, j) \leftarrow w * wordSim(s_i, t_j)$
10 $+ (1 - w) * contextSim$
11 Linearize and sort Ψ in decreasing order of $\Phi(i, j)$
12 $A \leftarrow \varnothing$
13 **for** $(i, j) \in \Psi$ **do**
14 **if** $\neg\exists l : (i, l) \in A$
15 $\wedge \neg\exists k : (k, j) \in A$ **then**
16 $A \leftarrow A \cup \{(i, j)\}$

(line 8), and the alignment score is a weighted sum of word similarity and contextual similarity (lines 9, 10). The alignment score is then used to make one-to-one word alignment decisions (lines 11-16). Considering textual neighbors as weaker sources of evidence, we do not align the neighbors.

Note that in *cwTextAlign* we also align semantically similar content word pairs (s_i, t_j) with no contextual similarities if no pairs (s_k, t_j) or (s_i, t_l) exist with a higher alignment score. This is a consequence of our observation of the MSR alignment *dev* data, where we find that more often than not content words are inherently sufficiently meaningful to be aligned even in the absence of contextual evidence when there are no competing pairs.

The content word alignment module is thus itself a pipeline of *cwDepAlign* and *cwTextAlign*.

3.3.4 Stop Words

We follow the contextual evidence-based approach to align stop words as well, some of which get aligned as part of word sequence alignment as discussed in Section 3.3.1, and neighbor alignment as discussed in Section 3.3.3. For the rest we utilize dependencies and textual neighborhoods as before, with three adjustments.

Algorithm 5: $align(S, T, EQ, w, \text{STOP})$

Input:
1. S, T: Sentences to be aligned
2. EQ: Dependency type equivalences (Table 1)
3. w: Weight of word similarity relative to contextual similarity
4. STOP: A set of stop words

Output: $A = \{(i, j)\}$: word index pairs of aligned words $\{(s_i, t_j)\}$ where $s_i \in S$ and $t_j \in T$

1 $A \leftarrow wsAlign(S, T)$
2 $A \leftarrow A \cup neAlign(S, T, EQ, A, w)$
3 $A \leftarrow A \cup cwDepAlign(S, T, EQ, A, w, \text{STOP})$
4 $A \leftarrow A \cup cwTextAlign(S, T, A, w, \text{STOP})$
5 $A \leftarrow A \cup swDepAlign(S, T, A, w, \text{STOP})$
6 $A \leftarrow A \cup swTextAlign(S, T, A, w, \text{STOP})$

Firstly, since stop word alignment is the last module in our pipeline, rather than considering all semantically similar word pairs for contextual similarity, we consider only aligned pairs. Secondly, since many stop words (e.g. determiners, modals) typically demonstrate little variation in the dependencies they engage in, we ignore type equivalences for stop words and implement only exact matching of dependencies. (Stop words in general can participate in equivalent dependencies, but we leave constructing a corresponding mapping for future work.) Finally, for textual neighborhood, we separately check alignments of the left and the right neighbors and aggregate the evidences to determine alignment – again due to the primarily syntactic nature of interaction of stop words with their neighbors.

Thus stop words are also aligned in a sequence of dependency and textual neighborhood-based alignments. We assume two corresponding modules named *swDepAlign* and *swTextAlign*, respectively.

3.3.5 The Algorithm

Our full alignment pipeline is shown as the method $align$ in Algorithm 5. Note that the strict order of the alignment modules limits the scope of downstream modules since each such module discards any word that has already been aligned by an earlier module

(this is accomplished via the variable A; the corresponding parameter in Algorithms 3 and 4 is A_E).

The rationales behind the specific order of the modules can now be explained: (1) *wsAlign* is a module with relatively higher precision, (2) it is convenient to align named entities before other content words to enable alignment of entity mentions of different lengths, (3) dependency-based evidence was observed to be more reliable (i.e. of higher precision) than textual evidence in the MSR alignment *dev* set, and (4) stop word alignments are dependent on existing content word alignments.

The aligner assumes two free parameters: $ppdbSim$ and w (in Algorithms 3 and 4). To determine their values, we exhaustively search through the two-dimensional space $(ppdbSim, w)$ for $ppdbSim, w \in \{0.1, ..., 0.9, 1\}$ and the combination $(0.9, 0.9)$ yields the best F_1 score for the MSR alignment *dev* set. We use these values for the aligner in all of its subsequent applications.

4 Evaluation

We evaluate the performance of our aligner both intrinsically and extrinsically on multiple corpora.

4.1 Intrinsic Evaluation

The MSR alignment dataset[2] (Brockett, 2007) was designed to train and directly evaluate automated aligners. Three annotators individually aligned words and phrases in 1600 pairs of *premise* and *hypothesis* sentences from the RTE2 challenge data (divided into *dev* and *test* sets, each consisting of 800 sentences). The dataset has subsequently been used to assess several top performing aligners (MacCartney et al., 2008; Thadani and McKeown, 2011; Yao et al., 2013a; Yao et al., 2013b). We use the *test* set for evaluation in the same manner as these studies: (a) we apply majority rule to select from the three sets of annotations for each sentence and discard three-way disagreements, (b) we evaluate only on the *sure* links (word pairs that annotators mentioned should certainly be aligned, as opposed to *possible* links).

We test the generalizability of the aligner by evaluating it, unchanged (i.e. with identical parameter values), on a second alignment corpus: the Edinburgh++[3] (Thadani et al., 2012) corpus. The *test* set consists of 306 pairs; each pair is aligned by at most two annotators and we adopt the random selection policy described in (Thadani et al., 2012) to resolve disagreements.

	System	P%	R%	F_1%	E%
MSR	MacCartney et al. (2008)	85.4	85.3	85.3	21.3
	Thadani & McKeown (2011)	89.5	86.2	87.8	33.0
	Yao et al. (2013a)	**93.7**	84.0	88.6	35.3
	Yao et al. (2013b)	92.1	82.8	86.8	29.1
	This Work	**93.7**	**89.8**	**91.7**	**43.8**
EDB++	Yao et al. (2013a)	91.3	82.0	86.4	15.0
	Yao et al. (2013b)	90.4	81.9	85.9	13.7
	This Work	**93.5**	**82.5**	**87.6**	**18.3**

Table 2: Results of intrinsic evaluation on two datasets

Table 2 shows the results. For each corpus, it shows *precision* (% alignments that matched with gold annotations), *recall* (% gold alignments discovered by the aligner), F_1 score and the percentage of sentences that received the exact gold alignments (denoted by E) from the aligner.

On the MSR test set, our aligner shows a 3.1% improvement in F_1 score over the previous best system (Yao et al., 2013a) with a 27.2% error reduction. Importantly, it demonstrates a considerable increase in recall without a loss of precision. The E score also increases as a consequence.

On the Edinburgh++ test set, our system achieves a 1.2% increase in F_1 score (an error reduction of 8.8%) over the previous best system (Yao et al., 2013a), with improvements in both precision and recall. This is a remarkable result that demonstrates the general applicability of the aligner, as no parameter tuning took place.

4.1.1 Ablation Test

We perform a set of ablation tests to assess the importance of the aligner's individual components. Each row of Table 3 beginning with (-) shows a feature excluded from the aligner and two associated sets of metrics, showing the performance of the resulting algorithm on the two alignment corpora.

Without a word similarity module, recall drops as would be expected. Without contextual evidence (word sequences, dependencies and textual neighbors) precision drops considerably and recall also falls. Without dependencies, the aligner still gives

[2] http://www.cs.biu.ac.il/ nlp/files/RTE_2006_Aligned.zip

[3] http://www.ling.ohio-state.edu/~scott/#edinburgh-plusplus

	MSR			EDB++		
Feature	**P%**	**R%**	**F_1%**	**P%**	**R%**	**F_1%**
Original	93.7	89.8	91.7	93.5	82.5	87.6
(-) Word Similarity	95.2	86.3	90.5	95.1	77.3	85.3
(-) Contextual Evidence	81.3	86.0	83.6	86.4	80.6	83.4
(-) Dependencies	94.2	88.8	91.4	93.8	81.3	87.1
(-) Text Neighborhood	85.5	90.4	87.9	90.4	84.3	87.2
(-) Stop Words	94.2	88.3	91.2	92.2	80.0	85.7

Table 3: Ablation test results

state-of-the-art results, which points to the possibility of a very fast yet high-performance aligner. Without evidence from textual neighbors, however, the precision of the aligner suffers badly. Textual neighbors find alignments across different lexical categories, a type of alignment that is currently not supported by our dependency equivalences. Extending the set of dependency equivalences might alleviate this issue. Finally, without stop words (i.e. while aligning content words only) recall drops just a little for each corpus, which is expected as content words form the vast majority of non-identical word alignments.

4.2 Extrinsic Evaluation

We extrinsically evaluate our system on textual similarity identification and paraphrase detection. Here we discuss each task and the results of evaluation.

4.2.1 Semantic Textual Similarity

Given two short input text fragments (commonly sentences), the goal of this task is to identify the degree to which the two fragments are semantically similar. The *SEM 2013 STS task (Agirre et al., 2013) assessed a number of STS systems on four test datasets by comparing their output scores to human annotations. Pearson correlation coefficient with human annotations was computed individually for each test set and a weighted sum of the correlations was used as the final evaluation metric (the weight for each dataset was proportional to its size).

We apply our aligner to the task by aligning each sentence pair and taking the proportion of content words aligned in the two sentences (by normalizing with the harmonic mean of their number of content words) as a proxy of their semantic similarity. Only three of the four STS 2013 datasets are freely available[4] (*headlines*, *OnWN*, and *FNWN*), which we use for our experiments (leaving out the *SMT* dataset).

[4]http://ixa2.si.ehu.es/sts/

System	**Correl.%**	**Rank**
Han et al. (2013)	**73.7**	1 (original)
JacanaAlign	46.2	66
This Work	67.2	7

Table 4: Extrinsic evaluation on STS 2013 data

These three sets contain 1500 annotated sentence pairs in total.

Table 4 shows the results. The first row shows the performance of the top system in the task. With a direct application of our aligner (no parameter tuning), our STS algorithm acheives a 67.15% weighted correlation, which would earn it the 7th rank among 90 participating systems. Considering the fact that alignment is one of many components of STS, this result is truly promising.

For comparison, we also evaluate the previous best aligner named JacanaAlign (Yao et al., 2013a) on STS 2013 data (the JacanaAlign public release[5] is used, which is a version of the original aligner with extra lexical resources). We apply three different values derived from its output as proxies of semantic similarity: a) aligned content word proportion, b) the Viterbi decoding score, and c) the normalized decoding score. Of the three, (b) gives the best results, which we show in row 2 of Table 4. Our aligner outperforms JacanaAlign by a large margin.

4.2.2 Paraphrase Identification

The goal of paraphrase identification is to decide if two sentences have the same meaning. The output is a yes/no decision instead of a real-valued similarity score as in STS. We use the MSR paraphrase corpus[6] (4076 *dev* pairs, 1725 *test* pairs) (Dolan et al., 2004) to evaluate our aligner and compare with other aligners. Following earlier work (MacCartney et al., 2008; Yao et al., 2013b), we use a normalized alignment score of the two sentences to make a decision based on a threshold which we set using the *dev* set. Alignments with a higher-than-threshold score are taken to be paraphrases and the rest non-paraphrases.

Again, this is an oversimplified application of the aligner, even more so than in STS, since a small change in linguistic properties of two sentences (e.g. polarity or modality) can turn them into non-

[5]https://code.google.com/p/jacana/

[6]http://research.microsoft.com/en-us/downloads/607d14d9-20cd-47e3-85bc-a2f65cd28042/

System	Acc.%	P%	R%	F_1%
Madnani et al. (2012)	**77.4**	***79.0***	*89.9*	**84.1**
Yao et al. (2013a)	70.0	72.6	88.1	*79.6*
Yao et al. (2013b)	68.1	68.6	**95.8**	*79.9*
This Work	73.4	76.6	86.4	81.2

Table 5: Extrinsic evaluation on MSR paraphrase data

paraphrases despite having a high degree of alignment. So the aligner was not expected to demonstrate state-of-the-art performance, but still it gets close as shown in Table 5. The first column shows the accuracy of each system in classifying the input sentences into one of two classes: *true* (paraphrases) and *false* (non-paraphrases). The rest of the columns show the performance of the system for the true class in terms of precision, recall, and F_1 score. *Italicized* numbers represent scores that were not reported by the authors of the corresponding papers, but have been reconstructed from the reported data (and hence are likely to have small precision errors).

The first row shows the best performance by any system on the test set to the best of our knowledge. The next two rows show the performances of two state-of-the-art aligners (performances of both systems were reported in (Yao et al., 2013b)). The last row shows the performance of our aligner. Although it does worse than the best paraphrase system, it outperforms the other aligners.

5 Discussion

Our experiments reveal that a word aligner based on simple measures of lexical and contextual similarity can demonstrate state-of-the-art accuracy. However, as alignment is frequently a component of larger tasks, high accuracy alone is not always sufficient. Other dimensions of an aligner's usability include speed, consumption of computing resources, replicability, and generalizability to different applications. Our design goals include achieving a balance among such multifarious and conflicting goals.

A speed advantage of our aligner stems from formulating the problem as one-to-one word alignment and thus avoiding an expensive decoding phase. The presence of multiple phases is offset by discarding already aligned words in subsequent phases. The use of PPDB as the only (hashable) word similarity resource helps in reducing latency as well as space requirements. As shown in Section 4.1.1, further speedup could be achieved with only a small performance degradation by considering only the textual neighborhood as source of contextual evidence.

However, the two major goals that we believe the aligner achieves to the greatest extent are replicability and generalizability. The easy replicability of the aligner stems from its use of only basic and frequently used NLP modules (a lemmatizer, a POS tagger, an NER module, and a dependency parser: all available as part of the Stanford CoreNLP suite[7]; we use a Python wrapper[8]) and a single word similarity resource (PPDB).

We experimentally show that the aligner can be successfully applied to different alignment datasets as well as multiple end tasks. We believe a design characteristic that enhances the generalizability of the aligner is its minimal dependence on the MSR alignment training data, which originates from a textual entailment corpus having unique properties such as disparities in the lengths of the input sentences and a directional nature of their relationship (i.e., the premise implying the hypothesis, but not vice versa). A related potential reason is the symmetry of the aligner's output (caused by its assumption of no directionality) – the fact that it outputs the same set of alignments regardless of the order of the input sentences, in contrast to most existing aligners.

Major limitations of the aligner include the inability to align phrases, including multiword expressions. It is incapable of capturing and exploiting long distance dependencies among words (e.g. coreferences). No word similarity resource is perfect and PPDB is no exception, therefore certain word alignments also remain undetected.

6 Conclusions

We show how contextual evidence can be used to construct a monolingual word aligner with certain desired properties, including state-of-the-art accuracy, easy replicability, and high generalizability. Some potential avenues for future work include: allowing phrase-level alignment via phrasal similarity resources (e.g. the phrasal paraphrases of PPDB), including other sources of similarity such as vector space models or WordNet relations, expanding the set

[7] http://nlp.stanford.edu/downloads/corenlp.shtml

[8] https://github.com/dasmith/stanford-corenlp-python

of dependency equivalences and/or using semantic role equivalences, and formulating our alignment algorithm as objective optimization rather than greedy search.

The aligner is available for download at `https://github.com/ma-sultan/monolingual-word-aligner`.

Acknowledgments

This material is based in part upon work supported by the National Science Foundation under Grant Numbers EHR/0835393 and EHR/0835381. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic Textual Similarity. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics.* Association for Computational Linguistics, 32-43.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with Bilingual Parallel Corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics.* Association for Computational Linguistics, 597-604.

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. 2012. UKP: computing semantic textual similarity by combining multiple content similarity measures. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics.* Association for Computational Linguistics, 435-440.

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The Second PASCAL Recognising Textual Entailment Challenge. In *Proceedings of The Second PASCAL Recognising Textual Entailment Challenge.*

Chris Brockett. 2007. Aligning the RTE 2006 Corpus. Technical Report MSR-TR-2007-77, Microsoft Research.

Nathanael Chambers, Daniel Cer, Trond Grenager, David Hall, Chloe Kiddon, Bill MacCartney, Marie-Catherine de Marneffe, Daniel Ramage, Eric Yeh, and Christopher D Manning. 2007. Learning alignments and leveraging natural logic. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing* Association for Computational Linguistics, 165-170.

Ming-Wei Chang, Dan Goldwasser, Dan Roth, and Vivek Srikumar. 2010. Discriminative Learning over Constrained Latent Representations. In *Proceedings of the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* Association for Computational Linguistics, 429-437.

Dipanjan Das and Noah A. Smith. 2009. Paraphrase Identication as Probabilistic Quasi-Synchronous Recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP.* Association for Computational Linguistics, 468-476.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of the International Conference on Language Resources and Evaluation.* 449-454.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. Stanford typed dependencies manual. Technical Report, Stanford University.

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.* Association for Computational Linguistics, 350-356.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics.* Association for Computational Linguistics, 363-370.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics, 758-764.

Lushan Han, Abhay Kashyap, Tim Finin, James Mayeld, and Jonathan Weese. 2013. UMBC EBIQUITY-CORE: Semantic Textual Similarity Systems. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics, Volume 1.* Association for Computational Linguistics, 44-52.

Andrew Hickl and Jeremy Bensley. 2007. A Discourse Commitment-Based Framework for Recognizing Textual Entailment. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing.* Association for Computational Linguistics, 171-176.

Andrew Hickl, Jeremy Bensley, John Williams, Kirk Roberts, Bryan Rink, and Ying Shi. 2006. Recognizing Textual Entailment with LCCs GROUNDHOG

System. In *Proceedings of the Second PASCAL Challenges Workshop on Recognizing Textual Entailment.*

Milen Kouylekov and Bernardo Magnini. 2005. Recognizing textual entailment with tree edit distance algorithms. In *Proceedings of the PASCAL Challenges Workshop: Recognising Textual Entailment Challenge* 17-20.

Bill MacCartney, Michel Galley, and Christopher D. Manning. 2008. A Phrase-Based Alignment Model for Natural Language Inference. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing.* Association for Computational Linguistics, 802-811.

Nitin Madnani, Joel Tetreault, and Martin Chodorow. 2012. Re-examining Machine Translation Metrics for Paraphrase Identification. In *Proceedings of 2012 Conference of the North American Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics, 182-190.

Kapil Thadani and Kathleen McKeown. 2011. Optimal and Syntactically-Informed Decoding for Monolingual Phrase-Based Alignment. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies.* Association for Computational Linguistics, 254-259.

Kapil Thadani, Scott Martin, and Michael White. 2012. A Joint Phrasal and Dependency Model for Paraphrase Alignment. In *Proceedings of COLING 2012: Posters.* 1229-1238.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003 Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics, 173-180.

Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. 2013a. A Lightweight and High Performance Monolingual Word Aligner. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics.* Association for Computational Linguistics, 702-707.

Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. 2013b. Semi-Markov Phrase-based Monolingual Alignment. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing.* Association for Computational Linguistics, 590-600.

Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. 2013c. Answer Extraction as Sequence Tagging with Tree Edit Distance. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics, 858-867.