Consistency Checking for Treebank Alignment

Markus Dickinson
Indiana University
md7@indiana.edu

Yvonne Samuelsson
Stockholm University
yvonne.samuelsson@ling.su.se

Abstract

This paper explores ways to detect errors in aligned corpora, using very little technology. In the first method, applicable to any aligned corpus, we consider alignment as a string-to-string mapping. Treating the target string as a label, we examine each source string to find inconsistencies in alignment. Despite setting up the problem on a par with grammatical annotation, we demonstrate crucial differences in sorting errors from legitimate variations. The second method examines phrase nodes which are predicted to be aligned, based on the alignment of their yields. Both methods are effective in complementary ways.

1 Introduction

Parallel corpora—texts and their translations—have become essential in the development of machine translation (MT) systems. Alignment quality is crucial to these corpora; as Tiedemann (2003) states, "[t]he most important feature of texts and their translations is the correspondence between source and target segments" (p. 2). While being useful for translation studies and foreign language pedagogy (see, e.g., Botley et al., 2000; McEnery and Wilson, 1996), PARALLEL TREEBANKS—syntactically-annotated parallel corpora—offer additional useful information for machine translation, cross-language information retrieval, and word-sense disambiguation (see, e.g., Tiedemann, 2003),

While high-quality alignments are desirable, even gold standard annotation can contain annotation errors. For other forms of linguistic annotation, the presence of errors has been shown to create various problems, from unreliable training and evaluation of NLP technology (e.g., Padro and Marquez, 1998) to low precision and recall of queries for already rare linguistic phenomena (e.g., Meurers and Müller, 2008). Even a small number of errors can have a significant impact on the uses of linguistic annotation, e.g., changing the assessment of parsers (e.g., Habash et al., 2007). One could remove potentially unfavorable sentence pairs when training a statistical MT system, to avoid incorrect word alignments (Okita, 2009), but this removes all relevant data from those sentences and does not help evaluation.

We thus focus on detecting errors in the annotation of alignments. Annotation error detection has been explored for part-of-speech (POS) annotation (e.g., Loftsson, 2009) and syntactic annotation (e.g., Ule and Simov, 2004; Dickinson and Meurers, 2005), but there have been few, if any, attempts to develop general approaches to error detection for aligned corpora. Alignments are different in nature, as the annotation does not introduce abstract categories such as POS, but relies upon defining translation units with equivalent meanings.

We use the idea that variation in annotation can indicate errors (section 2), for consistency checking of alignments, as detailed in section 3. In section 4, we outline language-independent heuristics to sort true ambiguities from errors, and evaluate them on a parallel treebank in section 5. In section 6 we turn to a complementary method, exploiting compositional properties of aligned treebanks, to align more nodes. The methods are simple, effective, and applicable to any aligned treebank. As far as we know, this is the first attempt to thoroughly investigate and empirically verify error detection methods for aligned corpora.

Proceedings of the Fourth Linguistic Annotation Workshop, ACL 2010, pages 38–46,
Uppsala, Sweden, 15-16 July 2010.

2 Background

2.1 Variation N-gram Method

As a starting point for an error detection method for aligned corpora, we use the variation n-gram approach for syntactic annotation (Dickinson and Meurers, 2003, 2005). The approach is based on detecting strings which occur multiple times in the corpus with varying annotation, the so-called VARIATION NUCLEI. The nucleus with repeated surrounding context is referred to as a VARIATION n-GRAM. The basic heuristic for detecting annotation errors requires one word of recurring context on each side of the nucleus, which is sufficient for detecting errors in grammatical annotation with high precision (Dickinson, 2008).

The approach detects bracketing and labeling errors in constituency annotation. For example, the variation nucleus *last month* occurs once in the Penn Treebank (Taylor et al., 2003) with the label NP and once as a non-constituent, handled through a special label NIL. As a labeling error example, *next Tuesday* occurs three times, twice as NP and once as PP (Dickinson and Meurers, 2003). The method works for discontinuous constituency annotation (Dickinson and Meurers, 2005), allowing one to apply it to alignments, which may span over several words.

2.2 Parallel Treebank Consistency Checking

For the experiments in this paper we will use the SMULTRON parallel treebank of Swedish, German, and English (Gustafson-Čapková et al., 2007), containing syntactic annotation and alignment on both word and phrase levels.[1] Additionally, alignments are marked as showing either an EXACT or a FUZZY (approximate) equivalence.

Corpora with alignments often have undergone some error-checking. Previous consistency checks for SMULTRON, for example, consisted of running one script for comparing differences in length between the source and target language items, and one script for comparing alignment labels, to detect variation between EXACT and FUZZY links. For example, the pair *and* (English) and *samt* (German, 'together with') had 20 FUZZY matches and 1 (erroneous) EXACT match. Such methods are limited, in that they do not, e.g., handle missing alignments.

The TreeAligner[2] tool for annotating and querying aligned parallel treebanks (Volk et al., 2007) employs its own consistency checking, recently developed by Torsten Marek. One method uses 2×2 contingency tables over words, looking, e.g., at the word-word or POS-POS combinations, pinpointing anomalous translation equivalents. While potentially effective, this does not address the use of alignments in context, i.e., when we might expect to see a rare translation.

A second, more treebank-specific method checks for so-called *branch link locality*: if two nodes are aligned, any node dominating one of them can only be aligned to a node dominating the other one. While this constraint can flag erroneous links, it too does not address missing alignments. The two methods we propose in this paper address these limitations and can be used to complement this work. Furthermore, these methods have not been evaluated, whereas we evaluate our methods.

3 Consistency of Alignment

To adapt the variation n-gram method and determine whether strings in a corpus are consistently aligned, we must: 1) define the units of data we expect to be consistently annotated (this section), and 2) define which information effectively identifies the erroneous cases (section 4).

3.1 Units of Data

Alignment relates words in a source language and words in a target language, potentially mediated by phrase nodes. Following the variation n-gram method, we define the units of data, i.e., the variation nuclei, as strings. Then, we break the problem into two different source-to-target mappings, mapping a source variation nucleus to a target language label. With a German-English aligned corpus, for example, we look for the consistency of aligning German words to their English counterparts and separately examine the consistency of aligning English words with their German "labels." Because a translated word can be used in different parts of a sentence, we also normalize all target labels into lower-case, preventing variation between, e.g., *the* and *The*.

[1] SMULTRON is freely available for research purposes, see `http://www.cl.uzh.ch/kitt/smultron/`.

[2] `http://www.cl.uzh.ch/kitt/treealigner`

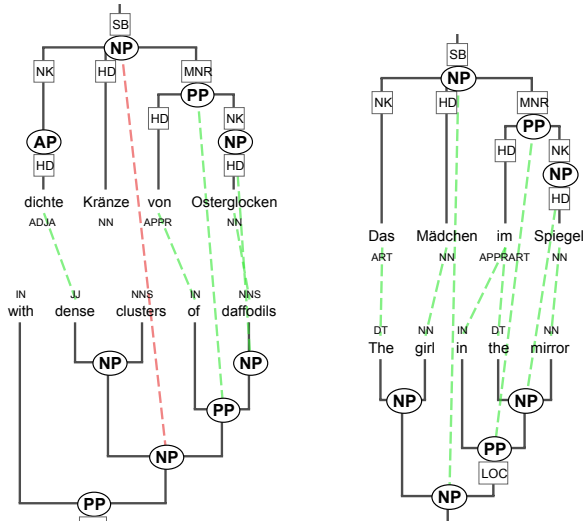

Figure 1: Word and phrase alignments span the same string on the left, but not on the right.

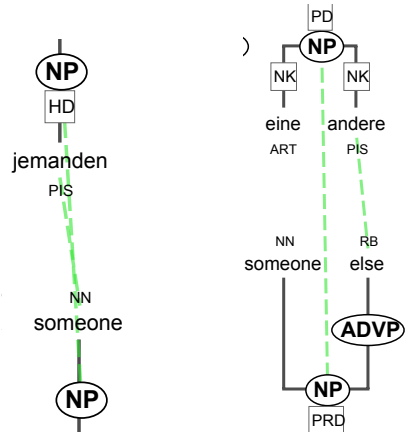

Figure 2: The word *someone* aligned as a phrase on the left, but not a phrase by itself on the right.

Although alignment maps strings to strings for this method, complications arise when mediated by phrase nodes: if a phrase node spans over only one word, it could have two distinct mappings, one as a word and one as a phrase, which may or may not result in the same yield. Figure 1 illustrates this. On the left side, *Osterglocken* is aligned to *daffodils* at the word level, and the same string is aligned on the phrase level (NP to NP). In contrast, on the right side, the word *Spiegel* is aligned to the word *mirror*, while at the phrase level, *Spiegel* (NP) is aligned to *the mirror* (NP). As word and phrase level strings can behave differently, we split error detection into word-level and phrase-level methods, to avoid unnecessary variation. By splitting the problem first into different source-to-target mappings and then into words and phrases, we do not have to change the underlying way of finding consistency.

Multiple Alignment The mapping between source strings and target labels handles n-to-m alignments. For example, if *Gärten* maps to *the gardens*, *the* and *gardens* is considered one string. Likewise, in the opposite direction, *the gardens* maps as a unit to *Gärten*, even if discontinuous.

Unary Branches With syntactic annotation, unary branches present a potential difficulty, in that a single string could have more than one label, violating the assumption that the string-to-label mapping is a function. For example, in Penn Treebank-style annotation, an NP node can dominate a QP (quantifier phrase) node via a unary branch. Thus, an annotator could (likely erroneously) assign different alignments to each phrasal node, one for the NP and one for the QP, resulting in different target labels.

We handle all the (source) unary branch alignments as a conjunction of possibilities, ordered from top to bottom. Just as the syntactic structure can be relabeled as NP/QP (Dickinson and Meurers, 2003), we can relabel a string as, e.g., *the man*/*man*. If different unary nodes result in the same string (*the man*/*the man*), we combine them (*the man*). Note that unary branches are unproblematic in the target language since they always yield the same string, i.e., are still one label.

3.2 Consistency and Completeness

Error detection for syntactic annotation finds inconsistencies in constituent labeling (e.g., NP vs. QP) and inconsistencies in bracketing (e.g., NP vs. NIL). Likewise, we can distinguish inconsistency in labeling (different translations) from inconsistency in alignment (aligned/unaligned). Detecting inconsistency in alignment deals with the completeness of the annotation, by using the label NIL for unaligned strings.

We use the method from Dickinson and Meurers (2005) to generate NILs, but using NIL for unaligned strings is too coarse-grained for phrase-level alignment. A string mapping to NIL might be a phrase which has no alignment, or it might

not be a phrase and thus could not possibly have an alignment. Thus, we create NIL-C as a new label, indicating a constituent with no alignment, differing from NIL strings which do not even form a phrase. For example, on the left side of Figure 2, the string *someone* aligns to *jemanden* on the phrase level. On the right side of Figure 2, the string *someone* by itself does not constitute a phrase (even though the alignment in this instance is correct) and is labeled NIL. If there were instances of *someone* as an NP with no alignment, this would be NIL-C. NIL-C cases seem to be useful for inconsistency detection, as we expect consistency for items annotated as a phrase.

3.3 Alignment Types

Aligned corpora often specify additional information about each alignment, e.g., a "sure" or "possible" alignment (Och and Ney, 2003). In SMULTRON, for instance, an EXACT alignment means that the strings are considered direct translation equivalents outside the current sentence context, whereas a FUZZY one is not as strict an equivalent. For example, *something* in English EXACT-aligns with *etwas* in German. However, if *something* and *irgend etwas* ('something or other') are constituents on the phrase level, <*something*, *irgend etwas*> is an acceptable alignment (since the corpus aligns as much as possible), but is FUZZY.

Since EXACT alignments are the ones we expect to consistently align with the same string across the corpus, we attach information about the alignment type to each corpus position. This can be used to filter out variations involving, e.g., FUZZY alignments (see section 4.4). When multiple alignments form a single variation nucleus, there could be different types of alignment for each link, e.g., *dog* EXACT-aligning and *the* FUZZY-aligning with *Hund*. We did not observe this, but one can easily allow for a mixed type (EXACT-FUZZY).

3.4 Algorithm

The algorithm first splits the data into appropriate units (SL=source language, TL=target language):

1. Divide the alignments into two SL-to-TL mappings.
2. Divide each SL-to-TL alignment set into word-level and phrase-level alignments.

For each of the four sets of alignments:

1. Map each string in SL with an alignment to a label
   - Label = <(lower-cased) TL translation, EXACT|FUZZY|EXACT-FUZZY>
   - (For phrases) Constituent phrases with no alignment are given the special label, NIL-C.
   - (For phrases) Constituent phrases which are unary branches are given a single, normalized label representing all target strings.
2. Generate NIL alignments for string tokens which occur in SL, but have no alignment to TL, using the method described in Dickinson and Meurers (2005).
3. Find SL strings which have variation in labeling.
4. Filter the variations from step 3, based on likelihood of being an error (see section 4).

4 Identifying Inconsistent Alignments

As words and phrases have acceptable variants for translation, the method in section 3 will lead to detecting acceptable variations. We use several heuristics to filter the set of variations.

4.1 NIL-only Variation

As discussed in section 3.2, we use the label NIL-C to refer to syntactic constituents which do not receive an alignment, while NIL refers to non-constituent strings without an alignment. A string which varies between NIL and NIL-C, then, is not really varying in its alignment—i.e., it is always unaligned. We thus remove cases varying only between NIL and NIL-C.

4.2 Context-based Filtering

The variation n-gram method has generally relied upon immediate lexical context around the variation nucleus, in order to sort errors from ambiguities (Dickinson, 2008). However, while useful for grammatical annotation, it is not clear how useful the surrounding context is for translation tasks, given the wide range of possible translations for the same context. Further, requiring identical context around source words is very strict, leading to sparse data problems, and it ignores alignment-specific information (see sections 4.3 and 4.4).

We test three different notions of context. Matching the variation n-gram method, we first employ a filter identifying those nuclei which share the "shortest" identical context, i.e., one word of context on every side of a nucleus. Secondly, we relax this to require only one word of

context, on either the left or right side. Finally, we require no identical context in the source language and rely only on other filters. For example, with the nucleus *come* in the context *Where does the world come from*, the first notion requires *world come from* to recur, the second either *world come* or *come from*, and the third only requires that the nucleus itself recur (*come*).

4.3 Target Language Filtering

Because translation is open-ended, there can be different translations in a corpus. We want to filter out cases where there is variation in alignment stemming from multiple translation possibilities. We implement a TARGET LANGUAGE FILTER, which keeps only the variations where the target words are present in the same sentence. If word x is sometimes aligned to y_1 and sometimes to y_2, and word y_2 occurs in at least one sentence where y_1 is the chosen target, then we keep the variation. If y_1 and y_2 do not occur in any of the same sentences, we remove the variation: given the translations, there is no possibility of having the same alignment.

This also works for NIL labels, given sentence alignments.[3] For NILs, the check is in only one direction: the aligned sentence must contain the target string used as the label elsewhere in the corpus. For instance, the word *All* aligns once with *alle* and twice with NIL. We check the two NIL cases to see whether one of them contains *alle*.

Sentences which are completely unaligned lead to NILs for every word and phrase, and we always keep the variation. In practice, the issue of having no alignment should be handled separately.

4.4 Alignment Type Filtering

A final filter relies on alignment type information. Namely, the FUZZY label already indicates that the alignment is not perfect, i.e., not necessarily applicable in other contexts. For example, the English word *dead* FUZZY-aligns with the German *verschwunden* ('gone, missing'), the best translation in its context. In another part of the corpus, *dead* EXACT-aligns with *leblosen* ('lifeless'). While this is variation between *verschwunden* and *leblosen*, the presence of the FUZZY label alerts us to the fact that it should vary with another word. The ALIGNMENT TYPE FILTER removes cases varying between one EXACT label and one or more FUZZY labels.

	word	phrase
all	540	251
oneword	340	182
shortest	96	21
all-TL	194	140
oneword-TL	130	94
shortest-TL	30	16

Table 1: Number of variations across contexts

5 Evaluation

Evaluation was done for English to German on half of SMULTRON (the part taken from the novel *Sophie's World*), with approximately 7500 words from each language and 7600 alignments (roughly 4800 word-level and 2800 phrase-level). Basic statistics are in Table 1. We filter based on the target language (*TL*) and provide three different contextual definitions: no context, i.e., all variations (*all*); one word of context on the left *or* right (*oneword*); and one word of context on the left *and* right, i.e., the shortest surrounding context (*shortest*). The filters reduce the number of variations, with a dramatic loss for the shortest contexts.

A main question concerns the impact of the filtering conditions on error detection. To gauge this, we randomly selected 50 (*all*) variations for the word level and 50 for the phrase level, each corresponding to just under 400 corpus instances. The variations were checked manually to see which were true variations and which were errors.

We report the effect of different filters on precision and recall in Table 2, where *recall* is with respect to the *all* condition.[4] Adding too much lexical context in the source language (i.e., the *shortest* conditions) results in too low a recall to be practically effective. Using one word of context on either side has higher recall, but the precision is no better than using no source language context at all. What seems to be most effective is to only use the target language filter (*all-TL*). Here, we find higher precision—higher than any source language filter—and the recall is respectable.

[3] In SMULTRON, sentence alignments are not given directly, but can be deduced from the set of word alignments.

[4] Future work should test for recall of all alignment errors, by first manually checking a small section of the corpus.

	Word				Phrase			
	Cases	Errors	P	R	Cases	Errors	P	R
all	50	17	34%	100%	50	15	30%	100%
oneword	33	12	36%	71%	33	8	24%	53%
shortest	8	2	25%	12%	4	1	25%	7%
all-TL	20	11	55%	65%	27	12	44%	80%
oneword-TL	15	6	40%	35%	14	7	50%	47%
shortest-TL	2	1	50%	6%	3	1	33%	7%

Table 2: Error precision and recall

TL filter An advantage of the target language filter is its ability to handle lexical (e.g., case) variations. One example of this is the English phrase *a dog*, which varies between German *einem Hund* (dative singular), *einen Hund* (accusative singular) and *Hunde* (accusative plural). Similar to using lower-case labels, one could map strings to canonical forms. However, the target language filter naturally eliminates such unwanted variation, without any language-specific information, because the other forms do not appear across sentences.

Several of the variations which the target language filter incorrectly removes would, once the error is fixed, still have variation. As an example, consider *cat*, which varies between *Katze* (5 tokens) and NIL (2 tokens). In one of the NIL cases, the word needs to be FUZZY-aligned with the German *Tigerkatze*. The variation points out the error, but there would still be variation (between *Katze*, *Tigerkatze*, and NIL) after correction. This shows the limitation of the heuristic in identifying the required non-exact alignments.

Another case the filter misses is the variation nucleus *heard*, which varies between *gehört* (2 tokens) and *hören* (1 token). In this case, one of the instances of <*heard*, *gehört*> should be <*heard*, *gehört hatte*>. Note that here the erroneous case is not variation-based at all; it is a problem with the label *gehört*. What is needed is a method to detect more translation possibilities.

As an example of a problem for phrases, consider the variation for the nucleus *end* with 5 instances of NIL and 1 of *ein Ende*. In one NIL instance, the proper alignment should be <*the end*, *Ende*>, with a longer source string. Since the target label is *Ende* and not *ein Ende*, the filter removes this variation. One might explore more fuzzily matching NIL strings, so that *Ende* matches with *ein Ende*. We explore a different method for phrases next, which deals with some of these NIL cases.

6 A Complementary Method

Although it works for any type of aligned corpus, the string-based variation method of detecting errors is limited in the types of errors it can detect. There might be ways to generalize the variation n-gram method (cf. Dickinson, 2008), but this does not exploit properties inherent to aligned treebanks. We pursue a complementary approach, as this can fill in some gaps a string-based method cannot deal with (cf. Loftsson, 2009).

6.1 Phrase Alignment Based on Word Links

Using the existing word alignments, we can search for missing or erroneous phrase alignments. If the words dominated by a phrase are aligned, the phrases generally should be, too (cf. Lavie et al., 2008). We take the yield of a constituent in one side of a corpus, find the word alignments of this yield, and use these alignments to predict a phrasal alignment for the constituent. If the predicted alignment is not annotated, it is flagged as a possible error. This is similar to the branch link locality of the TreeAligner (see section 2.2), but here as a prediction, rather than a restriction, of alignment.

For example, consider the English VP *choose her own friends* in (1). Most of the words are aligned to words within *Ihre Freunde vielleicht wählen* ('possibly choose her friends'), with no alignment to words outside of this German VP. We want to predict that the phrases be aligned.

(1) a. [$_{VP}$ choose$_1$ her$_2$ own friends$_3$]
b. [$_{VP}$ Ihre$_2$ Freunde$_3$ vielleicht wählen$_1$]

The algorithm works as follows:

1. For every phrasal node s in the source treebank:
   (a) Predict a target phrase node t to align with, where t could be non-alignment (NIL):

i. Obtain the yield (i.e., child nodes) of the phrase node s: s_1, ... s_n.

ii. Obtain the alignments for each child node s_i, resulting in a set of child nodes in the target language (t_1, ... t_m).

iii. Store every mother node t' covering all the target child nodes, i.e., all $<s, t'>$ pairs.

(b) If a predicted alignment ($<s, t'>$) is not in the set of actual alignments ($<s, t>$), add it to the set of potential alignments, $A_{S \mapsto T}$.

i. For nodes which are predicted to have non-alignment (but are actually aligned), output them to a separate file.

2. Perform step 1 with the source and target reversed, thereby generating both $A_{S \mapsto T}$ and $A_{T \mapsto S}$.

3. Intersect $A_{S \mapsto T}$ and $A_{T \mapsto S}$, to obtain the set of predicted phrasal alignments not currently aligned.

The main idea in 1a is to find the children of a source node and their alignments and then obtain the target nodes which have all of these aligned nodes as children. A node covering all these target children is a plausible candidate for alignment.

Consider example (2). Within the 8-word English ADVP (*almost twice* . . .), there are six words which align to words in the corresponding German sentence, all under the same NP.[5] It does not matter that some words are unaligned; the fact that the English ADVP and the German NP cover basically the same set of words suggests that the phrases should be aligned, as is the case here.

(2) a. Sophie lived on$_2$ [$_{NP_1}$ the$_2$ outskirts$_3$ of a$_4$ sprawling$_{5*}$ suburb$_{6*}$] and had [$_{ADVP}$ almost$_7$ twice$_8$ as$_9$ far$_{10}$ to school as$_{11}$ Joanna$_{12*}$] .

b. Sophie wohnte am$_2$ [$_{NP_1}$ Ende$_3$ eines$_4$ ausgedehnten$_{5*}$ Viertels$_{6*}$ mit Einfamilienhäusern] und hatte [$_{NP}$ einen fast$_7$ doppelt$_8$ so$_9$ langen$_{10}$ Schulweg wie$_{11}$ Jorunn$_{12*}$] .

The prediction of an aligned node in 1a allows for multiple possibilities: in 1aiii, we only check that a mother node t' covers all the target children, disregarding extra children, since translations can contain extra words. In general, many such dominating nodes exist, and most are poor candidates for alignment of the node in question. This is the reason for the bidirectional check in steps 2 and 3.

For example, in (3), we correctly predict alignment between the NP dominating *you* in English and the NP dominating *man* in German. From the word alignment, we generate a list of mother nodes of *man* as potential alignments for the *you* NP. Two of these (six) nodes are shown in (3b). In the other direction, there are eight nodes containing *you*; two are shown in (3a). These are the predicted alignment nodes for the NP dominating *man*. In either direction, this overgenerates; the intersection, however, only contains alignment between the lowest NPs.

(3) a. But it 's just as impossible to realize [$_S$ [$_{NP}$ **you**$_1$] have to die without thinking how incredibly amazing it is to be alive] .

b. [$_S$ Und es ist genauso unmöglich , darüber nachzudenken , dass [$_{NP}$ **man**$_1$] sterben muss , ohne zugleich daran zu denken , wie phantastisch das Leben ist .]

While generally effective, certain predictions are less likely to be errors. In figure 3, for example, the sentence pair is an entire rephrasing; *<her, ihr>* is the only word alignment. For each phrasal node in the SL, the method only requires that all its words be aligned with the words under the TL node. Thus, the English PP *on her*, the VP *had just been dumped on her*, and the two VPs in between are predicted as possible alignments with the German VP *ihr einfach in die Wiege gelegt worden* or its immediate VP daughter: they all have *her* and *ihr* aligned, and no contradicting alignments. Sparse word alignments lead to multiple possible phrase alignments. After intersecting, we mark cases with more than one predicted source or target phrase and do not evaluate them.

If in step 1aiii, no target mother (t') exists, but there is alignment in the corpus, then in step 1bi, we output predicted non-alignment. In Example (2), for instance, the English NP *the outskirts of a sprawling suburb* is (incorrectly) predicted to have no alignment, although most words align to words within the same German NP. This prediction arises because *the* aligns to a word (*am*) outside of the German NP, due to *am* being a contraction of the preposition *an* and the article *dem*, (cf. *on* and *the*, respectively). The method for predicting phrase alignments, however, relies upon words being within the constituent. We thus conclude that: 1) the cases in step 1bi are unlikely to be errors, and 2) there are types of alignments which we simply will not find, a problem also for automatic alignment based on similar assumptions (e.g., Zhechev and Way, 2008). In (2), for instance, were there not already alignment between

[5] FUZZY labels are marked by an asterisk, but are not used.

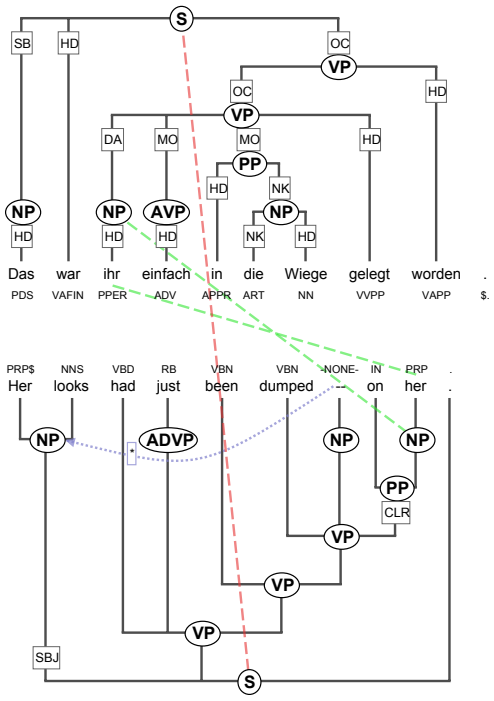

Figure 3: A sentence with minimal alignment

the NPs, we would not predict it.

6.2 Evaluation

The method returns 318 cases, in addition to 135 cases with multiple source/target phrases and 104 predicted non-alignments. To evaluate, we sampled 55 of the 318 flagged phrases and found that 25 should have been aligned as suggested. 21 of the phrases have zero difference in length between source and target, while 34 have differences of up to 9 tokens. Of the phrases with zero-length difference, 18 should have been aligned (precision=85.7%), while only 7 with length differences should have been aligned. This is in line with previous findings that length difference can help predict alignment (cf., e.g., Gale and Church, 1993). About half of all phrase pairs that should be aligned should be EXACT, regardless of the length difference.

The method is good at predicting the alignment of one-word phrases, e.g., pronouns, as in (3). Of the 11 suggested alignments where both source and target have a length of 1, all were correct suggestions. This is not surprising, since all words under the phrases are (trivially) aligned. Although shorter phrases with short length differences generally means a higher rate of correct suggestions, we do not want to filter out items based on phrase length, since there are outliers that are correct suggestions, e.g., phrase pairs with lengths of 15 and 13 (difference=2) or 31 and 36 (difference=5). It is worth noting that checking the suggestions took very little time.

7 Summary and Outlook

This paper explores two simple, language-independent ways to detect errors in aligned corpora. In the first method, applicable to any aligned corpus, we consider alignment as a string-to-string mapping, where a string could be the yield of a phrase. Treating the target string as a label, we find inconsistencies in the labeling of each source string. Despite setting the problem up in a similar way to grammatical annotation, we also demonstrated that new heuristics are needed to sort errors. The second method examines phrase nodes which are predicted to be aligned, based on the alignment of their yields. Both methods are effective, in complementary ways, and can be used to suggest alignments for annotators or to suggest revisions for incorrect alignments.

The wide range of possible translations and the linguistic information which goes into them indicate that there should be other ways of finding errors. One possibility is to use more abstract source or target language representations, such as POS, to overcome the limitations of string-based methods. This will likely also be a useful avenue to explore for language pairs more dissimilar than English and German. By investigating different ways to ensure alignment consistency, one can begin to provide insights into automatic alignment (Zhechev and Way, 2008). Additionally, by correcting the errors, one can determine the effect on machine translation evaluation.

Acknowledgments

We would like to thank Martin Volk and Thorsten Marek for useful discussion and feedback of earlier versions of this paper and three anonymous reviewers for their comments.

References

Botley, S. P., McEnery, A. M., and Wilson, A., editors (2000). *Multilingual Corpora in Teaching and Research*. Rodopi, Amsterdam, Atlanta GA.

Dickinson, M. (2008). Representations for category disambiguation. In *Proceedings of COLING-08*, pages 201–208, Manchester.

Dickinson, M. and Meurers, W. D. (2003). Detecting inconsistencies in treebanks. In *Proceedings of TLT-03*, pages 45–56, Växjö, Sweden.

Dickinson, M. and Meurers, W. D. (2005). Detecting errors in discontinuous structural annotation. In *Proceedings of ACL-05*, pages 322–329.

Gale, W. A. and Church, K. W. (1993). A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102.

Gustafson-Čapková, S., Samuelsson, Y., and Volk, M. (2007). SMULTRON (version 1.0) - The Stockholm MULtilingual parallel TReebank. www.ling.su.se/dali/research/smultron/index.htm.

Habash, N., Gabbard, R., Rambow, O., Kulick, S., and Marcus, M. (2007). Determining case in Arabic: Learning complex linguistic behavior requires complex linguistic features. In *Proceedings of EMNLP-CoNLL-07*, pages 1084–1092.

Lavie, A., Parlikar, A., and Ambati, V. (2008). Syntax-driven learning of sub-sentential translation equivalents and translation rules from parsed parallel corpora. In *Proceedings of the ACL-08: HLT Second Workshop on Syntax and Structure in Statistical Translation (SSST-2)*, pages 87–95, Columbus, OH.

Loftsson, H. (2009). Correcting a POS-tagged corpus using three complementary methods. In *Proceedings of EACL-09*, pages 523–531, Athens, Greece.

McEnery, T. and Wilson, A. (1996). *Corpus Linguistics*. Edinburgh University Press, Edinburgh.

Meurers, D. and Müller, S. (2008). Corpora and syntax (article 44). In Lüdeling, A. and Kytö, M., editors, *Corpus Linguistics. An International Handbook*, Handbooks of Linguistics and Communication Science. Mouton de Gruyter, Berlin.

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

Okita, T. (2009). Data cleaning for word alignment. In *Proceedings of the ACL-IJCNLP 2009 Student Research Workshop*, pages 72–80, Suntec, Singapore.

Padro, L. and Marquez, L. (1998). On the evaluation and comparison of taggers: the effect of noise in testing corpora. In *Proceedings of ACL-COLING-98*, pages 997–1002, San Francisco, California.

Taylor, A., Marcus, M., and Santorini, B. (2003). The penn treebank: An overview. In Abeillé, A., editor, *Treebanks: Building and using syntactically annotated corpora*, chapter 1, pages 5–22. Kluwer, Dordrecht.

Tiedemann, J. (2003). *Recycling Translations - Extraction of Lexical Data from Parallel Corpora and their Application in Natural Language Processing*. PhD thesis, Uppsala university.

Ule, T. and Simov, K. (2004). Unexpected productions may well be errors. In *Proceedings of LREC-04*, Lisbon, Portugal.

Volk, M., Lundborg, J., and Mettler, M. (2007). A search tool for parallel treebanks. In *Proceedings of the Linguistic Annotation Workshop (LAW) at ACL*, pages 85–92, Prague, Czech Republic. Association for Computational Linguistics.

Zhechev, V. and Way, A. (2008). Automatic generation of parallel treebanks. In *Proceedings of Coling 2008*, pages 1105–1112, Manchester, UK.