An Efficient Unified Extraction Algorithm for Bilingual Data

Christoph Tillmann[1], *Sanjika Hewavitharana*[2]

[1]IBM T.J. Watson Research Center, Yorktown Heights, NY 10598
[2]Carnegie Mellon University, Pittsburgh, PA 15213
ctill@us.ibm.com, sanjika@cs.cmu.edu

Abstract

The paper presents a unified algorithm for aligning sentences with their translations in bilingual data. The sentence alignment problem is handled as a large-scale pattern recognition problem similar to the task of finding the word sequence that corresponds to an acoustic input signal in isolated word automatic speech recognition (ASR). The algorithm gains efficiency from related work on dynamic programming (DP) search for speech recognition ([1]): a stack-based search is parametrized in a novel way, such that the unified algorithm can be used on various types of data that have been previously handled by separate implementations: the extracted text chunk pairs can be either sub-sentential pairs, one-to-one, or many-to-many sentence-level pairs. The one-stage search algorithm is carried out in a single run over the data. With the help of a unified beam-search candidate pruning, the algorithm is very efficient: it avoids any document-level pre-filtering and uses less restrictive sentence-level filtering. Results are presented on a Russian-English and a Spanish-English extraction task. Based on a simple word-based scoring model, text chunk pairs are extracted out of several trillion candidates.
Index Terms: statistical machine translation, bilingual data extraction, ASR search .

1. Introduction

This paper presents an algorithm for extracting translation correspondences from non-sentence-aligned bilingual data. Originally, such algorithms have been proposed for parallel corpora which can be aligned largely monotonically with no reorderings and only limited insertions and deletions between the two sentence collections [2, 3]. More recent work has been focused on extracting parallel data from comparable noisy data. Typically, the data is pre-filtered at the document level [4, 5], i.e. the data comes annotated with document boundaries and matching document pairs are pre-selected based on IR (Information-Retrieval) criteria. One-to-one sentence-level pairs are then selected from within the matching document pairs. Here, a sentence pair is only considered a possible match, if both sentences have been published within a publication date window of $\pm n$ days, e.g. $n = 3$. In addition, a sentence length filter is applied: source and target sentence length can differ by at most a factor 2. The sentence alignment can be further used to extract sub-sentential fragment pairs [5, 6]. In the current paper, we present a single unified search algorithm that can be used flexibly on all the different extraction tasks handled by the various algorithms cited above. Translation pair extraction is handled as a chunk-alignment problem: no document-level pre-filtering is used and the data is processed directly at the sentence level. The resulting search space can be huge: for a given source sentence the set of possible target candidates might contain 100 000 sentences. Our sentence alignment algorithm can be compared directly to isolated word speech recognition where a sequence of acoustic vectors is matched against all the target words in the recognizer's vocabulary. The novel algorithm uses techniques from large-scale DP (dynamic programming) algorithms in speech recognition [1]: the search space is dynamically constructed, and just two stacks are used to process the parallel data at the word level. To carry out the sentence alignment, the algorithm uses a lexical scoring function based on the so-called IBM Model-1 [7] that can be computed efficiently [8]. The generalized chunk pair alignment problem is handled as a *pattern recognition* problem. The source document is the *input pattern* that is assumed to be decomposed into text chunks of source words. Those source chunks are aligned to *reference patterns* that are target word chunks. In particular, the search in comparable data can be compared directly to a pattern recognition formulation of the ASR search problem [1]. The source chunks correspond to words in the acoustic input signal. Each document position within a chunk corresponds to an acoustic time frame in the ASR search. Each time frame in the ASR search is represented by an acoustic vector that is matched against the reference patterns. Similarly, each source chunk word position is represented by a Model-1 based matching score with respect to all the target candidates. Because the chunk alignment problem considers text chunk pairs of different granularity, it is a more general pattern search algorithm than the ASR search in [1] which considers only one-to-one word-level pairs.

Section 2 presents the extraction task for various data conditions as an alignment problem. Section 3 shows how these types are handled jointly within a unified stack-based algorithm with special beam-search pruning thresholds. In Section 4, we show experimental results. Section 5 compares our algorithm to previous work in the literature and discusses future work.

2. Alignment Problem and Scoring

The following three search types are handled jointly by the unified algorithm (an illustration is given in Fig. 1):

I. Monotone search: In [2, 3], the sentence alignment search is restricted to monotone alignments which results in an efficient search algorithm. This base search problem is handled by the current algorithm efficiently and a full monotone search at the sentence level can be carried out. A word-level monotone search is used to compute a sub-sentential fragment alignment.

II. Sentence-level re-ordering search: A restricted sentence level re-ordering search is introduced. It uses a target coverage vector to handle some sentence-order differences in mostly monotone data. This type is related to the way a decoder for statistical machine translation (SMT) handles word-level reordering [9]. Even though the search space is restricted by the use of a coverage vector, we do not use a distortion model between neighbor text chunks.

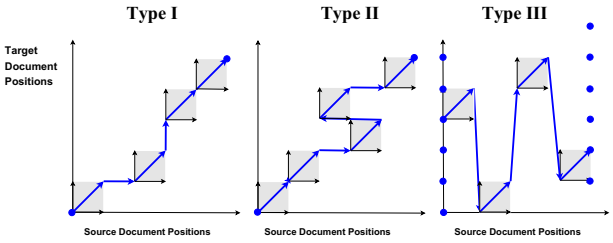

Figure 1: The search algorithm finds a shortest path through the bitext space. The unified algorithm handles three alignment types: Type I search is related to connected word speech recognition and Type III search is related to isolated word speech recognition.

III. Comparable data search: When searching for sentence-level pairs in comparable data, no monotonicity assumption is made for source and target sentences. Typically, only one-to-one sentence pairs are extracted [4, 5, 8]. A search for sub-sentential fragment pairs is handled as a combination of the search for alignment types I and III as shown below.

The different search types extract bilingual pairs of text chunks: a chunk consists of a sequence of source or target words [10]. Chunk pairs are either sub-sentential fragment pairs, one-to-one, or many-to-many sentence-level pairs. A chunk pair corresponds to a *link* in the bitext:

$$[\, j_{beg}(l)\ ,\ i_{beg}(l)\ ,\ j_{end}(l)\ ,\ i_{end}(l)\,]\ ,$$

where the link l starts at source position $j_{beg}(l) \in \{1, \cdots, \mathcal{J}\}$ and target position $i_{beg}(l) \in \{1, \cdots, \mathcal{I}\}$ ($j_{end}(l)$ and $i_{end}(l)$ are restricted accordingly). Here, $\mathcal{I}$ and $\mathcal{J}$ are the number of words in the target document and source document, respectively. In order to be able to extract sub-sentential as well as sentence-level links with a single implementation, links are defined at the word level. To do so efficiently, the algorithm maintains a mapping index: for each source sentence s_m and target sentence t_n, we compute its word-level start and end position. For a parallel sentence collection, these positions are computed up-front before the search. For the experiments in Section 4, we consider 1-to-1, 1-to-4, and 4-to-4 type links at the sentence level. For the 1-to-4 type links, a single sentence is aligned to up to 4 sentences in the other language. Special 'null' links for insertions and deletions are included. Similarly, for the 4-to-4-type links up to 4 source and target sentence are aligned to each other. In addition, we search for sub-sentential links of up to 10 source and target words. Link types can be handled flexibly at run time: they correspond to state expansions in the dynamically constructed search space.

We search for the highest scoring link sequence l_1^N of N links (the number N is unknown beforehand), which covers all the words in the source document:

$$c(l_1^N) \quad = \quad \max_{l_1^N} \sum_{i=1}^{N} \varrho(\, S(l_i), T(l_i)\,), \qquad (1)$$

where $\varrho(S,T)$ is the *intra*-chunk cost for matching source chunk $S(l)$ with target chunk $T(l)$ as defined below. Here, $S(l)$ is the source chunk covered by link l and $T(l)$ is defined accordingly. Currently, the *inter*-chunk cost between neighbor chunk pairs is zero. The intra-chunk cost $\varrho(S,T)$ is based on the so-called IBM Model-1 [7] . The Model-1 is trained on some parallel data available for a language pair, i.e. the data used to train the baseline systems in Section 4. The Model-1 score $\varrho(S,T)$ (log probabilities) is computed for two directions: source to target and target to source, and the two scores are normalized by sentence length: this way the score is independent of the source and target chunk length (for a definition of $\varrho(S,T)$, see [8]). A single threshold θ is used to select all those link pairs whose score is above it. For Russian-English, we use a threshold of $\theta = -6.4$, and for Spanish-English a threshold of $\theta = -5.6$. They are chosen on development sets with respect to classification accuracy. $\varrho(S,T)$ can be computed very efficiently in an incremental fashion and has been shown to be effective in extracting one-to-one sentence-level chunk pairs [8].

3. Unified Search

This section presents the unified search algorithm. For all three search types, the algorithm finds a global path that corresponds to a link sequence that covers all source document words. The target coverage varies based on the different search types. The search is carried out in a single run over all source word positions j, where $1 \leq j \leq \mathcal{J}$ and all the different link types are handled uniformly across the different search types. The algorithm maintains two state lists: the list of currently active hypotheses Γ (at position j) , and the list of newly generated hypotheses Γ' (at position $j+1$). A search state σ is defined as follows:

$$\sigma = [\, l\ ;\ d\,]\ , \qquad (2)$$

where l is a word-level link and d is the partial matching score. This simple state definition is sufficient since there are no long range dependencies between links other than the path coverage restrictions for the different search types. During search we distinguish 'complete' from 'incomplete' states. Complete states correspond to links whose partial score computation has finished covering all source and target positions for the latest link,i.e. whose link end position equals the current source position j: $j_{end}(l) = j$. DP recombination is used to reduce the number of complete search states, i.e. for a monotone search at each source position j, we keep the highest scoring hypothesis for each target position i. The partial link sequences that correspond to the highest scoring states after recombination are extended by adding an additional link. A link sequence ending at a grid-point (j,i) can be extended by new links l' covering additional source and target positions, i.e. links for which $j_{beg}(l') = j+1$. The recombination and link extension procedures are based on list-processing functions that are similar to those used for DP-based speech recognition [1] and phrase-based SMT decoding [9] algorithms. Here, the two-stack implementation uses only constant memory with respect to the source document length. The scoring part of the algorithm is identical for all three search types. The score of a given link is computed incrementally on a word-by-word basis. While covering an additional source position j, the target coverage is computed proportionally [11]. Based on the source and target coverage, an incremental Model-1 score $\Delta_l(j)$ is computed. For all search types, the algorithm uses special source null-coverage links for uncovered source chunks. The null-coverage score is based on a default constant which is chosen with respect to minimal Model-1 probabilities. In addition, many-to-many null coverage links are also handled. Even though the algorithm computes the highest scoring global link sequence l_1^N, each link l_i is scored separately based on the $\varrho(S,T)$ score, e.g. a link l on the global path is only selected if $\varrho(l) > \theta$ locally. An important component of this memory efficient implementation is the so-called trace-back array which is implemented as a linear

data structure ([1]). Complete states that pass the recombination step are written into this array. We can recover the partial link sequence corresponding to a state by following the sequence of back-pointers in the trace-back array.

3.1. Search Type Details

For the Type I search, the algorithm uses dynamic programming to search over an exponential number of chunk alignments efficiently, and a full search can be carried out. The above two-stack implementation needs a memory of $2 \cdot \mathcal{T}$. In comparison, a standard implementation would need a memory of $\mathcal{S} \cdot \mathcal{T}$, where $\mathcal{S}$ and $\mathcal{T}$ are the number of source sentences and target sentences, respectively. For this monotone search, target null-coverage links are also needed in order to make sure that a global path that covers an entire document / sentence pair can be computed. When searching for sub-sentential fragments, we pre-match candidates at the sentence-level and carry out a monotone Type I search within the candidate sentences. The Type II search uses a target coverage vector $\mathcal{C}$. It is used to keep track of target sentences that have already been aligned. The use of the target coverage vector is similar to the use of a source word coverage vector in DP-based SMT decoders [9]. Here, we require that each target sentence is covered *at most* once (not *exactly* once), i.e. some target sentence are left uncovered. Since target documents may consist of thousands of sentences, the permissible coverage vectors need to be restricted. Based on the source sentence position s, we compute a restricted target coverage vector: a window of ± 10 target sentence positions relative to the bitext diagonal is considered. For the Type III search, we search over a large number of candidates: there is no monotone alignment restriction and each of the alignment link types may start at any target sentence position. Since there might be hundreds of thousands of candidates for a given source sentence, a Type III search can only be carried out in connection with the candidate pruning described in Section 3.2. The unified algorithm is capable of extracting many-to-many links as well as one-to-one links within a single run over the data. In comparison, algorithms in the literature carry out a search only for 1-to-1 sentence-level links.

3.2. Candidate Pruning

For the sub-sentential Type I and comparable data Type III searches, we cannot carry out a full search over the bilingual data. The set of links l considered during search must be restricted appropriately. To do so, we modify work on extracting one-to-one sentence-level translation pairs from comparable data [8]. During the left-to-right run over the source positions j, whenever we reach the starting position of a sentence S in the source document, we compute the following candidate set of sentence-level 1-to-1-type links l_i:

$$\Theta(S) = \{ l_i = (S, T_i) \mid T_i \text{ among the highest scoring } N \text{ links with score } \varrho(S, T_i) \}, \quad (3)$$

where N is a pruning threshold (for our experiment, we pick $N = 25$) and $\varrho(S, T_k)$ is the extraction score defined in Section 2. On the development sets used to determine the selection threshold θ, the true translation T for a source sentence S is in $\Theta(S)$ for 98 % of the source sentences. This candidate set can be computed very efficiently: the early stopping criterion and the efficient caching techniques introduced in [8] are modified accordingly. For the Type III search, the links considered are restricted based on the candidate set $\Theta(S)$ as follows: only those links l whose bitext start position is identical to one of the links l_i, i.e. $i_{beg}(l) = i_{beg}(l_i)$ and $j_{beg}(l) = j_{beg}(l_i)$ are considered, i.e. a many-to-many link is considered only if its lower left corner is part of the candidate set. For the Type I search with sub-sentential links, the candidate set $\Theta(S)$ restricts the number of links that are considered during search: the sub-sentential links l are covered entirely by one of the candidates l_i, i.e $l \subseteq l_i$. The candidate pruning step is related to similar pruning steps in SMT [9] and ASR decoders [12]. For a given source segment, i.e. a word or a phrase, only a restricted number of translation candidates are considered in the decoder. Apart from the candidate pruning step, no additional pruning is carried out currently, i.e. there is no global pruning based on the partial scores in the current beam at position j.

Table 1: BLEU scores for a Russian-English and a Spanish-English phrase-based systems trained on different data sets.

Type	link	# pairs	# src / # tgt words	Bleu
Russian-English				
Baseline		1 470 604	15.03 / 17.56 M	7.34
I	**1-1**	624 803	15.11 / 16.00 M	13.37
	1-4	399 792	22.27 / 21.71 M	12.59
	4-4	423 800	31.96 / 34.23 M	13.85
	Frag	1 520 465	41.31 / 42.42 M	14.36
II	**1-1**	1 058 316	24.92 / 27.33 M	14.30
	1-4	582 724	26.96 / 18.53 M	13.54
	4-4	338 756	27.63 / 23.08 M	13.56
III	**1-1**	1 477 759	33.84 / 36.47 M	14.76
	1-4	1 494 285	34.37 / 37.56 M	14.67
Spanish-English				
Baseline	-	1 825 709	48.27 / 46.04 M	42.42
I	Frag	1 684 384	44.41 / 36.87 M	45.26
III	**1-1**	1 510 243	34.53 / 29.45 M	45.01
	1-4	1 586 576	36.77 / 31.39 M	45.13

4. Experiments

In this section, we report extraction results for a Russian-English and a Spanish-English data collection. The probability models used to compute the chunk pair score $\varrho(S, T)$ are trained on some baseline data. For both languages, the baseline data comes from the news domain and technical manuals: the number of sentences and words for this baseline data is reported in Table 1. The Russian data consists of $16,346$ document pairs from the United Nations (UN). Since the sentence alignment for these document pairs is largely monotone, the Type I and II search types are applied. Here, the largest document pair consists of up to $10,000$ source / target sentences. For the Type II search, the beam might contain tens of thousands of states. Since the average sentence length is around 30 words, the overall search space reaches billions of states. Processing on such a document pair takes about 1 hour on a single processor. Another set of experiments is carried out on comparable English-Spanish data which comes from the Gigaword corpora for the years 2003-2008. No document-level alignment is provided for this data. A ± 3 day publication date filter and sentence-length filter are used: the length of source and target chunks may not differ more than by a factor two [4]. Already for the one-to-one link search, the search space is huge: link candidates are selected out of 1.08 trillion candidate pairs. There are about 20 million source sentences, and the average size of the

target candidate set is 50 000 sentences after the sentence-level filtering step. Since at each grid point $L = 16$ many-to-may candidates are considered, the overall candidate set of many-to-many links may reach several trillion chunk pairs. Processing the English-Spanish data takes about two days on a cluster with 300 processors. Processing the Russian-English UN data takes only a few hours on the same cluster where the un-pruned Type I monotone search runs fastest.

For each search type, we report the number of extracted translation correspondences as well as the number of source and target words in Table 1. We train translation systems based on the baseline data and each of the additional data sets based on the different search types. On the combined data, we train a phrase-based system and report its BLEU score on the test sets. Including the extracted parallel data leads to a significant improvement over the baseline system in all cases. For the two language pairs considered, one-to-one sentence level links already yield good extraction performance. On the UN data, three different types of sentence links are extracted: one-to-one, one-to-many, and many-to-many. On this data, the monotone Type I search is compared against the Type II re-ordering search. The Type II search improves BLEU by about 1 % for the **1-1** and **1-4** cases. Even on this data, Type I sub-sentential links and a Type III search actually outperform the monotonic search types: the document pairs are several thousand sentences long, and some sentence re-ordering cannot be handled by the Type II re-ordering window. On the comparable Spanish-English data, the fragment-based Type I search yields the best BLEU scores as shown in Table 1. Type III search with one-to-many links currently perform slightly better than the search with one-to-one links. No Type I and Type II sentence-level search results are presented, since the data is not aligned at the document level. The current approach has also been used successfully to extract Chinese-English and Arabic-English parallel data to build phrase-based SMT systems for translating broadcast news speech data. Details are omitted for brevity reasons.

5. Comparison and Future Work

[5, 6] extract fragments based on a two-step filtering pipeline: 1) matching document pairs are extracted based on an IR criterion, 2) from within the matching documents, sentence pairs are extracted based on a sentence-level matching criteria. Finally, sub-sentential fragments are extracted from within these sentence pairs. In this paper, we skip the document-level pre-filtering all-together, and formulate the sentence-level pre-filtering step as a special pruning step within the unified beam-search algorithm : even fragment extraction is carried out in a single run over the data. [11] presents a beam-search extraction algorithm for 1-to-1 sentence-level links from comparable data, i.e. a special case of the Type III search in this paper. The current paper extends this work in terms of a more flexible and unified search algorithm. In future, the algorithm might be evaluated in terms of segmentation accuracy on manually annotated data, and a more complex feature set might improve accuracy. Our link sequence search is similar to the pattern recognition approach presented in [13]. Their search is carried out by alternating between a generation and a recognition phase. Similarly, [14] sketches a two-pass algorithm that handles both parallel and comparable data, but it is tested on a small, artificial test set in terms of extraction accuracy. In comparison, the current paper presents a memory and run-time efficient search that processes comparable bilingual data sets at a huge scale based on ASR search techniques [1]. [10] describes an approach for segmenting and simultaneously aligning chunks of source and target texts (including sub-sentential chunks). Their paper considers two search strategies: a monotone DP-based search and an iterative binary search, but no results on comparable data are presented. In the current beam-search algorithm, the link-internal matching is deterministic: the partial target link coverage depends on the partial source coverage. Therefore, it can not be compared to the word-internal time warping in ASR decoders. In future, a fuzzy match between source and target sentences could be carried out, e.g. when searching for chunk pairs with gaps.

6. References

[1] H. Ney, "The Use of a One-stage Dynamic Programming Algorithm for Connected Word Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 2, pp. 263–271, 1984.

[2] W. A. Gale and K. W. Church, "A program for aligning sentences in bilingual corpora," in *Proc. of ACL'91*, Berkeley, CA, 1991, pp. 177–184.

[3] R. C. Moore, "Fast and accurate sentence alignment of bilingual data," in *Proc. of the 5h Conference of AMTA*, Tiburon, CA, 2002, pp. 135–144.

[4] D. S. Munteanu and D. Marcu, "Improving Machine Translation Performance by Exploiting Non-Parallel Corpora," *CL*, vol. 31, no. 4, pp. 477–504, 2005.

[5] C. Quirk, R. Udupa, and A. Menezes, "Generative Models of Noisy Translations with Applications to Parallel Fragment Extraction," in *Proc. of the MT Summit XI*, Copenhagen,Demark, September 2007, pp. 321–327.

[6] D. S. Munteanu and D. Marcu, "Extracting parallel sub-sentential fragments from non-parallel corpora," in *Proceedings of COLING/ACL'06*, Sydney, Australia, July 2006, pp. 81–88.

[7] P. F. Brown, V. J. Della Pietra, S. A. Della Pietra, and R. L. Mercer, "The Mathematics of Statistical Machine Translation: Parameter Estimation," *CL*, vol. 19, no. 2, pp. 263–311, 1993.

[8] C. Tillmann and J.-M. Xu, "A Simple Sentence-Level Extraction Algorithm for Comparable Data," in *Comp. Vol. of NAACL HLT 09*, Boulder, Colorado, June 2009, pp. 93–96.

[9] P. Koehn, "Pharaoh: a Beam Search Decoder for Phrase-Based SMT Models," in *Proceedings of AMTA'04*, Washington DC, September-October 2004.

[10] Y. Deng, S. Kumar, and W. Byrne, "Segmentation and Alignment of Parallel Text for Statistical Machine Translation," *Natural Language Engineering*, vol. 12, no. 4, pp. 1–26, 2006.

[11] C. Tillmann, "A Beam-Search Extraction Algorithm for Comparable Data," in *Short Papers ACL 09*, Singapore, August 2009, pp. 225–228.

[12] S. Ortmanns and H. Ney, "Progress in Dynamic Programming Search for LVCSR," *Proc. of the IEEE*, vol. 88, no. 8, pp. 1224–1240, 2000.

[13] I. D. Melamed, "Bitext Maps and Alignment via Pattern Recognition," *CL*, vol. 25, no. 1, pp. 107–130, 1999.

[14] C. Pike and I. D. Melamed, "An Automatic Filter for Non-Parallel Texts," in *The Comp. Volume of to the Proc. of ACL'04*, Barcelona, Spain, July 2004, pp. 114–117.