Constrained Hidden Markov Model for Bilingual Keyword Pairs Alignment

Denny Cahyadi   Fabien Cromieres   Sadao Kurohashi
Graduate School of Informatics, Kyoto University
Yoshida-Honmachi, Sakyo-ku, Kyoto, 606-8501, Japan
{denny,fabien}@nlp.ist.i.kyoto-u.ac.jp, kuro@i.kyoto-u.ac.jp

Abstract
Bilingual terminology dictionaries are resources of much practical importance in many application of bilingual NLP. Because technical terminology can be both very specific and rapidly evolving, it can however be difficult to obtain dictionaries with good coverage. Mining automatically such terminology from technical documents is therefore an attractive possibility. With this goal in mind, and following some previous works, we devise an algorithm that is efficient at aligning the bilingual keyword list of scientific papers. Our results show that our approach can extract bilingual terms with very good precision and recall.

Keywords: Hidden Markov Model, lexicon extraction, word alignment, alignment model.

1 Introduction

Bilingual terminology dictionaries can be critically useful for many practical tasks such as Machine Translation of technical documents, cross-language Information Retrieval or simply as a resource to human translators. Bilingual dictionaries of technical terms with good coverage and quality can however be difficult to obtain, both due to the high domain-specificity of most technical terms and the continuous creation of new terms as new research is being conducted and new technics are being developed.

Because of this, some research has been done to extract automatically such bilingual dictionaries from technical documents. This paper will try to further this research, in particular by following a previously proposed idea of aligning the keywords list of technical and scientific documents.

We propose an approach to adapt existing word-alignment algorithm to the task of keyword-list alignment. This is done by enforcing constraints on the keyword boundaries and by using a different distortion model for the keywords and the words within each keyword. In particular, we show that our modified version of the HMM alignment algorithm of (Vogel et al., 1996) perform better than both the original HMM algorithm and the commonly used word-aligner GIZA++ for the keyword alignment task.

2 Related works

If one has a large amount of technical documents in the relevant domain that are written in the two languages of interest, one can of course extract bilingual term pairs by running one of the many word-alignment algorithm that have already been proposed (such as (Brown et al., 1993) or (Vogel et al., 1996)). However, such bilingual collection of documents are not easily obtained. Besides, the word-alignment algorithms will not by themselves indicate which set of words represent a technical term. We will focus here on approaches that do not require such large bilingual collection of technical documents.

(Lin et al., 2008) reported experiments of extracting term pairs from part of document which contains parentheses. It is based on the fact that the English translation of a technical term is sometimes written inside a parenthesis next to it. Their method involves building a kind of parallel corpus by extracting the English terms in parenthesis and the words appearing before the parentheses, filtering the non-translation words, and then aligning the English terms with an unsupervised word alignment method.

(Nagata et al., 2001) extracted term pairs not only from parentheses, but also from glossaries and parallel paragraph. In their works, term pairs are filtered out if they are unlikely translation of each other. The likeliness is calculated based on their locations. Foreign and English terms which are appearing close to each other are considered to have higher translation likeliness. Their work is focused on extracting technical term pairs from Japanese documents.

(Ren et al., 2010) extracted bilingual technical term pairs from keyword section of abstracts of Chinese research papers. In their method, Chinese and English keywords are aligned monotonically[1] based on their writing order if the lists contain the same number of keywords. Otherwise, a seed bilingual dictionary is used to detect partial translation.

[1]Here and in the following, a monotonic alignment is one in which the first source keyword is aligned with the first target keyword, the second source keyword with the second target one, etc.

キーワード:	有機半導体レーザ、有機発光ダイオード、分布帰還レーザ、屈折率回折格子
Keyword:	organic semiconductor laser, distributed feedback, organic light-emitting diode, refractive index grating

Figure 1: An example of non-monotonic alignment. The second English keyword is a translation of third Japanese and the third English keyword is a partial translation of second Japanese keyword

Keywords which contain partial translation of each other are then aligned. Finally, keywords are filtered based on their inverse-domain score. This step is done to make sure that the keyword extracted are technical terms (not general term).

In this paper, we follow the idea of aligning keyword lists. However, instead of using a seed bilingual dictionary and making a strong assumption of monotonicity for the keyword order, we develop an unsupervised approach based on the adaptation of existing word-alignment algorithms to the specificities of the keyword-alignment task.

3 Keyword Alignment Model

3.1 Keyword Alignment Problem and Constraint

Figure 1 gives an example of the type of keyword list one can find in the description of a Japanese scientific paper. The keywords will often be written in the same order in both languages, but this is far from being systematic. Some keywords might also be mentioned in one language and not the other. For example, in our Japanese-English data (see section 4), around one third of the keyword lists exhibit different writing order or non-symmetric use of keywords.

The task of automatically aligning keyword lists presents some differences with the more common word-alignment of sentences. These differences can limit the efficiency of directly applying existing word alignment algorithm. Indeed, using such word-alignment algorithm lead to the question of what should be the elementary alignment unit (i.e. the "word" for the algorithm). It is natural to consider each individual word in the keyword list to be an alignment unit. But then, the alignment algorithm might produce alignments breaking the keyword boundaries: two words of the same source keywords might be aligned with two words in different target keywords. If on the other hand, one consider each keyword to be an alignment unit, data sparseness problem will appear: if "maximum entropy" is considered as a single unit, we cannot use the knowledge we may have of the translation of "maximum" or "entropy" to find the alignment.

An additional issue that will appear when using words as the alignment unit is that the order of the words in the keyword list is the result of two effects: the word order inside a keyword (that will follow the source or target language grammar), and the order of the keywords themselves (that will often -but not always- be similar between the source and target keyword list). Classic word-alignment algorithm are usually not designed to handle efficiently this kind of two-factor word distortion.

In the following, we will try to define an alignment model that address these issues. Many well-known word alignment algorithms ((Vogel et al., 1996), (Brown et al., 1993)) make use of a probabilistic distortion model that is estimating the probability of source and target words to be aligned depending on their position. We propose to augment these models with a 2-level distortion probability: a word-level distortion and a keyword-level distortion. This will serve two purpose: ensuring that the keyword boundaries are respected by the word-alignment, and modeling the two-factor word distortion.

We chose to focus on the HMM alignment model of (Vogel et al., 1996), as it is both simple (and thus easy to adapt) and efficient. However, the idea we develop here could be applied to most alignment model.

3.2 Standard HMM alignment

We briefly review the classic HMM alignment model of (Vogel et al., 1996). It is an asymmetric 1-to-n alignment model. e represents a source (or English) sentence, with e_i being the word at position i. Likewise, f is a target (or Foreign) sentence. The alignment variable a assign an English word (or a special Null word) to each foreign word: $a_i = j$ means that f_i is aligned with e_j. The probability of $P(a, f|e)$ is expressed as an HMM, where the hidden states are the source words e_j, the observed sequence is the target sentence f, and a specify a possible sequence of hidden states generating f:

$$P(a, f|e) = \prod_{j=1}^{|f|} P_{dist}(a_j|a_{j-1}) \cdot P_{trans}(f_j|e_{a_j}) \tag{1}$$

$P_{dist}(k|l)$, the state transition probability (or *distortion* probability), specify the probability of the alignment of two adjacent foreign words jumping from e_l to e_k. $P_{trans}(f_j|e_{a_j})$, the emission probability (or *translation* probability), specify the probability that f_j is the translation of e_{a_j}. This model can be trained on parallel sentences using the classic Baum-Welch algorithm.

3.3 Constrained HMM alignment

In the context of keyword lists, the position of a word is better represented as a pair of integer $i.j$, meaning that the word is at position j in keyword i. $a_{i.j} = k.l$ means that the source word at position $i.j$ is aligned with the target word at position $k.l$. Each keyword in both languages is augmented at the end with a special EOK (End of Keyword) word (we note $i.EOK$ the position of the EOK of keyword i). Each English keyword is also augmented with a special Null keyword.

We modify the HMM alignment model in order to solve the problems mentioned in section 3.1. We call this modified model *constrained HMM* since we enforce a constraint on the state transition so that every word alignment will respect the keyword boundaries. The constraint is expressed as the following: State transition to any arbitrary state is allowed only in the very beginning or after the aligner reach the end of a keyword (EOK) on the target side. In any other time, state transition is only allowed from the current state to another state that corresponds to the same keyword than the current state. In other words, we set $P_n(a_{j.n}|a_{i.m}) = 0$ if $m \neq EOK$ and $i \neq j$.

We decompose the state transition model into a keyword transition probability P_{distK} and a word transition probability P_{distW}. The initial probability of the first keyword and the first word of a keyword are given, respectively, by P_{initK} and P_{initW}.

The final state transition probability can then be expressed as follow (note the $j.n$ subscript of $P_{j.n}$ that shows that this transition probability will change depending on the position in a target keyword). :

$$P_{j.n}(a_{j.n} = k.l | a_{i.m} = k'.l') = \begin{cases} P_{initK}(k) \cdot P_{initW}(l) & \text{if } j = 1 \text{ and } n = 1 \\ P_{distK}(k|k') \cdot P_{initW}(l) & \text{if } n = 1 \\ P_{distW}(l|l') \cdot \delta_{kk'} & \text{otherwise} \end{cases} \tag{2}$$

In this formula, $\delta_{kk'}$ is a Kronecker delta (equal to 1 if $k = k'$, else 0). $i.m$ is always the position in the target keyword list just before $j.n$.

With this expression of the distortion, we have solved the issues we describes: the word alignment will respect keywords boundaries, and we model separately the keyword order and the word order inside the keywords. This lead to better alignment results, as we will see in section 4. The improvement can also be seen in the light of (Roweis, 2000), that shows that constraining a HMM in a way consistent with a task will lead to better a training. Note also that this model can be seen as aligning both words and keywords: although it gives a word alignment, the keyword boundary constraint means that this 1-to-n word alignment define unambiguously a 1-to-n keyword alignment.

3.4 Time-homogeneous implementation

This HMM model is not time homogeneous: the transition probability matrix is not the same depending on if we are in the middle or at the end of a target keyword. The Viterbi algorithm and the Baum-Welch training algorithm can perfectly be applied to such a HMM. However, many existing HMM library assume time homogeneity. If one want to use such library, it can be convenient to cast our model as a time-homogeneous HMM. This can be done at the price of doubling the number of states.

Firstly we want to know whether the current state corresponds to the same keyword than the previous state in the sequence. For this purpose, the number of hidden states is doubled. The originals are marked as e_i^{new} and their duplicates are marked as e_i^{cont}. State e_i^{new} is used if current state corresponds to different keyword than what the previous state corresponds to, otherwise e_i^{cont} is used.

By having these states, state sequence can be controlled easily by applying the following rules to the state transition model: 1) if current state is e_i^{new} then only transition to e_i^{cont} is allowed; 2) if current state is e_i^{cont}, then transition to any e_i^{cont} or EOK is allowed; 3) if current state is EOK, then only transition to e_i^{new} is allowed. For state emission model, the following rule are applied: visible state $f_{j.EOK}$ of foreign keyword f_j can only be emitted by hidden state $e_{i.EOK}$.

3.5 Variants

The natural way of finding the best alignment according to a trained HMM model is to use Viterbi decoding. However, (Liang et al., 2006) reports that one can get better results by computing the expected probability of each link and keeping those above a certain threshold. We can apply this idea here, although in our case we will want to use the expected probability of two keyword being aligned (easily computed from the expected probabilities of the word links).

Our HMM model is, like the original one, a 1-to-n alignment model. As is often done in such case, we can align each keyword list twice, with each language taken alternatively as the source language. We then combine the resulting alignments by intersecting their set of keyword links.

Although it is possible to start the training of the HMM model from some random model, it can also be beneficial to initialize the translation probabilities with those obtained from the simpler IBM Model 1 (Brown et al., 1993).

(Liang et al., 2006) show that it is beneficial for the final alignment quality to also train jointly the parameters of the HMM for each direction. We tried to do this, but did not observe any improvement in our results. We are not sure at this stage if this is due to the specificities of the task and the data, or to a subtle problem in our use of the training by agreement method.

Observing that more than half of all authors choose to write the keyword lists in both language in a perfectly parallel way (same keywords and same order), we also considered a model that would be a mixture of two HMM models: one such as the one we described in the previous section, and one that constrain the keyword order to be monotonic. We were thus hoping to model the idea that our data was actually generated by a mix of two sources: "organized" authors, that write perfectly parallel pairs of keywords lists, and "disorganized" authors, that take more freedom when they write these keyword lists. Unfortunately, this approach did not yield any improvement either. We are still unsure if the problem is with the basic assumptions or the way we designed and implemented the model.

Because the training by agreement and the mixture of model idea did not improve the results while complexifying the model and the implementation, we do not mention them in the experiment section.

4 Experiment

4.1 Data description

We conduct some experiments for extracting Japanese-English keyword pairs from Japanese research paper. We could obtain around 4 millions abstract of Japanese scientific papers, originating from CiNii[2] web portal. Only about 720k of them contain both English and Japanese keyword lists. We use these lists as the dataset for all experiments.

We create two sets of annotated keyword lists pairs to be used as test set and tuning set respectively. The main use of the tuning set is to set the threshold of alignment when we use the expected probability of each link instead of the Viterbi alignment (see section 3.5). Each set contains 100 keyword lists pairs taken randomly from the dataset. These pairs are aligned manually by two native Japanese speakers who are also fluent in English.

Segmentation for Japanese words is done using JUMAN (Kurohashi et al.). A keyword list usually contains about 2 to 20 keywords with average of 8 keywords. There are a total of 807 Japanese keywords in the test set.

[2]http://ci.nii.ac.jp

4.2 Baseline and Setup

Three different methods are used for baseline. As the first baseline, keywords are aligned monotonically in a way similar to (Ren et al., 2010) except we did not do the partial translation alignment as we want to consider unsupervised methods with no seed dictionary available. This baseline is called `mono-all` when we apply it to all keyword list, and `mono-same` when we apply it only to keyword list with the same length.

For the second baseline, alignment is done using GIZA++ alignment tool (Och and Ney, 2003) with its default setting. Alignment was done both way then merged by intersection[3]. With this method we conduct two experiments: using word (`wGIZA-i`) and keyword (`kGIZA-i`) as the alignment unit. When using word as the alignment unit, a keyword links is created for each word link. The `wGIZA-i` approach will tend to produce more links and possibly many-to-many keyword alignments, resulting in higher recall but lower precision than the `kGIZA-i` approach. For the third baseline, standard HMM (`sHMM`) is used for the alignment, with keyword as the alignment unit.

For the implementation of our constrained HMM, we conducted 4 experiments. First we applied constrained HMM in only one direction, with setup similar to `sHMM`, with and without initializing with the IBM Model1 parameters (`cHMM` and `cHMM-ibm`, see section 3.5). We then used bidirectional alignments using intersect method (`cHMM-ibm-i`).

Finally, we tried to create the alignment not by Viterbi decoding, but by thresholding the expected probability of the keyword links (`cHMM-ibm-i-t`, see section 3.5). Optimum threshold is determined using a separate tuning set.

4.3 Result

Method	Test set	Recall	Precision	F-score
`mono-same`	`test-set`	0.861	**0.968**	0.911
`mono-all`	`test-set`	0.890	0.742	0.809
`wGIZA-i`	`test-set`	0.970	0.867	0.915
`kGIZA-i`	`test-set`	0.922	0.906	0.914
`sHMM`	`test-set`	0.928	0.603	0.731
`cHMM`	`test-set`	0.968	0.629	0.763
`cHMM-ibm`	`test-set`	**0.977**	0.636	0.770
`cHMM-ibm-i`	`test-set`	0.956	0.898	0.926
`cHMM-ibm-i-t`	`test-set`	0.954	0.918	**0.936**
`mono-same`	`test-set-equal-length`	0.983	0.977	0.980
`cHMM-ibm-i-t`	`test-set-equal-length`	0.998	0.998	**0.998**

Table 1: Recall, Precision, and F-score of each method

The result of our experiments is shown in Table 1. They show several interesting things. One is that our modification to the HMM model yield improvement in both precision and recall over the standard version (`sHMM` vs `cHMM`). This appears to validate our view that modeling the specificities of the keywords lists is beneficial to alignment.

[3]The "grow-diag-final" heuristic is often used instead of a simple intersection. However, we found that in the case of keyword list, it was not performing better than a simple intersection.

Another interesting point is that the best version of our algorithm (`cHMM-ibm-i-t`) outperform all our baselines in term of F-measure by at least 2% absolute (corresponding to a relative error[4] reduction of 25%). It also gives the second best precision, behind the somewhat conservative heuristic `mono-same`. However, tuning the threshold used in `cHMM-ibm-i-t` for optimal precision rather than for optimal F-score would change that (see also section 4.4).

4.4 Closer comparison with the monotonic heuristics

Given the very good performance of `mono-same` and `mono-all` (with respect to their simplicity), we tried to compare the performance of our algorithm on the subset of data for which these heuristics should perform the best (and for which they are used in (Ren et al., 2010)): keyword lists with the same length.

We selected the 68 list pairs of our test set having the same number of English and Japanese keywords (`test-set-equal-length`). Of these, 63 are actually perfectly parallel keyword lists. The result (Table 1) shows that the monotonic heuristics are outperformed by our algorithm: it can not only detect all of the perfectly aligned pairs, but also correctly align 4 of the 5 non-monotonic lists.

4.5 Chinese-English

We intend to conduct a similar experiment for Chinese-English keyword list pairs. However we have so far only collected 70,000 such keyword lists. We report however our preliminary result on this smaller data set. Our alignment algorithm is still performing better than standard word-aligners such as Giza++ (by around 3.6% absolute F-score). However, it turns out that the simpler baseline of aligning monotonically each keywords (*mono-all*) gives even better results (2.4% absolute F-score difference) . This appears to be due to two reasons. One is that the smaller training data size has a negative impact on statistical aligners. The other is that there is much more perfectly parallel keyword lists in this dataset (more than 90%). The idea of a mixture of models (section 3.5) could therefore be effective on such data set.

Conclusion and perspectives

We considered the problem of aligning the bilingual keyword lists of scientific papers, with the goal of automatic bilingual terms extraction. We proposed to modify existing word-alignment algorithms in order to better take into account the specificities of this task. Our method show significant improvement over previous approaches and general word aligner, achieving better results than the often used Giza++ word aligner while having much less complexity. Besides the modifications we proposed could be applied to more complex alignment algorithm as well.

We need to investigate more, however, why some of our other tentative improvements (such as the training by agreement or the mixture of models) did not yield better results. We are also in the process of gathering additional Chinese-English data to perform more experiments in this language pair, and ultimately plan to combine and filter the two data sources to create a Japanese-Chinese dictionary of technical terms.

[4]Where we compute the error as 1 minus the F-score.

References

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19(2):263–311.

Kurohashi, S., Nakamura, T., Matsumoto, Y., and Nagao, M. Improvements of japanese morphological analyzer JUMAN. In *Proceedings of the International Workshop on Sharable Natural Language*, pages 22–28.

Liang, P., Taskar, B., and Klein, D. (2006). Alignment by agreement. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 104–111, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lin, D., Zhao, S., Durme, B. V., and Pasca, M. (2008). Mining parenthetical translations from the web by word alignment. In McKeown, K., Moore, J. D., Teufel, S., Allan, J., and Furui, S., editors, *ACL*, pages 994–1002. The Association for Computer Linguistics.

Nagata, M., Saito, T., and Suzuki, K. (2001). Using the web as a bilingual dictionary. In *Proceedings of the workshop on Data-driven methods in machine translation - Volume 14*, DMMT '01, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Ren, F., Zhu, J., and Wang, H. (2010). Web-based technical term translation pairs mining for patent document translation. In *Natural Language Processing and Knowledge Engineering (NLP-KE), 2010 International Conference on*, pages 1–8.

Roweis, S. (2000). Constrained hidden markov models. In *In Solla et al. (2000*, pages 782–788. MIT Press.

Vogel, S., Ney, H., and Tillmann, C. (1996). Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics - Volume 2*, COLING '96, pages 836–841, Stroudsburg, PA, USA. Association for Computational Linguistics.