TransSearch: from a bilingual concordancer to a translation finder

Julien Bourdaillet, Stéphane Huet, Philippe Langlais, Guy Lapalme

TransSearch: from a bilingual concordancer to a translation finder

Julien Bourdaillet, Stéphane Huet, Philippe Langlais
(felipe@iro.umontreal.ca) and Guy Lapalme
DIRO - Université de Montréal, C.P. 6128, succursale Centre-ville
H3C 3J7, Montréal, Québec, Canada

Abstract. As basic as bilingual concordancers may appear, they are some of the most widely used computer-assisted translation tools among professional translators. Nevertheless, they still do not benefit from recent breakthroughs in machine translation. This paper describes the improvement of the commercial bilingual concordancer TransSearch in order to embed a word alignment feature. The use of statistical word alignment methods allows the system to spot user query translations, and therefore turns it into a translation search engine. We describe several translation identification and post-processing algorithms that enhance the application. The excellent results obtained using a large translation memory consisting of 8.3 million sentence pairs are confirmed via human evaluation.

Keywords: Computer-assisted translation, Bilingual concordancer, Word alignment, Translation spotting, Evaluation, Filtering, Variant merging, Pseudo-relevance feedback

1. Introduction

During recent years, a tremendous amount of effort has been devoted to improving the current state of machine translation (MT). However, there is still room for improving computer-assisted translation (CAT) tools, which remain the most popular among professional translators. Some new approaches have been designed to ease the workflow of an end-user. For instance, TransType, a target-mediated interactive interface, assists translators in typing translations (Foster et al., 1997; Casacuberta et al., 2009) and its impact on the productivity of users has been extensively investigated (Macklovitch, 2006). More recently, Koehn and Haddow (2009) evaluated the use of MT methods inside a similar device.

Among CAT tools, bilingual concordancers play a prominent role. Given a query, they are used to retrieve source-target translation pairs in a translation memory (TM) whose source sentence contains the query. In contrast to bilingual dictionaries, these concordancers help in finding translations of multi-word expressions and in presenting various contexts of occurrence, thus providing ready-made solutions to all sorts of translation problems.

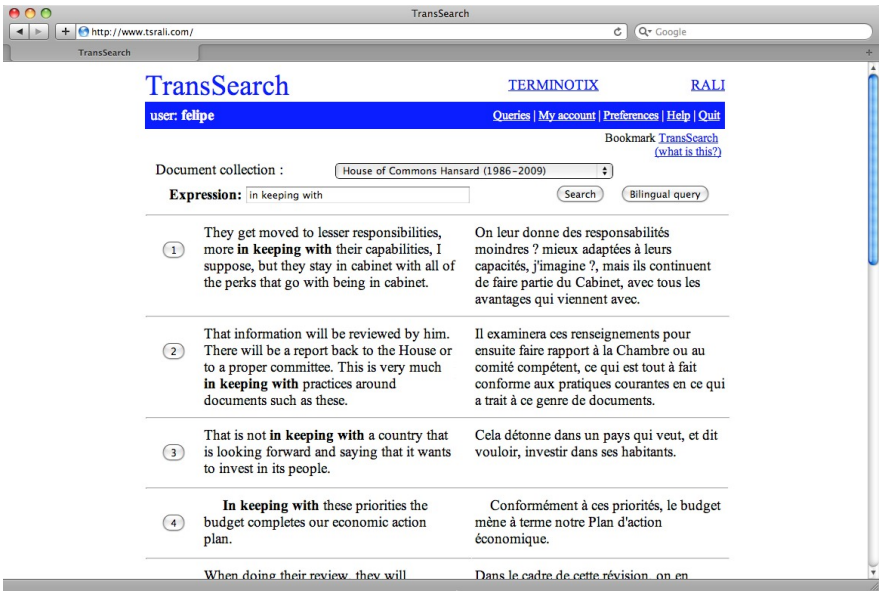

Figure 1. English sentences, along with their French translation, where the query `in keeping with` occurs in the current version of the bilingual concordancer TransSearch. The query is highlighted in the English sentences, but the user has to search its translations in the French sentences. This makes the discovery of the different translation possibilities more difficult (`adaptées à`, `conforme aux`, `détonne` and `conformément à` for the four displayed sentences), and forces the user to scroll down the page in order to read other sentences while searching for further translation material.

This paper describes the enhancement of the web-based bilingual concordancer TransSearch[1] (Macklovitch et al., 2000). While subscribers of the system are mainly professional translators, a recent study of their query logs exhibits that TransSearch is used to answer *difficult* translation problems (Macklovitch et al., 2008). Among the 7.2 million queries submitted to the system over a six-year period, 87% contain at least two words. Few queries are single words present in bilingual dictionaries, or bona fide terms which can be found in terminology banks. In fact, the users mainly search for: idiomatic expressions such as `in keeping with` (query frequency: 716 times), `in light of` (544 times), or `look forward to` (539 times); verbs or adjectives that govern a preposition such as `consistent with` (743 times), or `focus on` (472 times); and adverbials or adjectives whose precise translation depends on context like `as such` (1195 times), `at this time` (913 times), or `overarching` (417 times).

[1] TransSearch is available commercially at `http://www.tsrali.com`

Figure 1 shows the result page of the current version of the application, the user having submitted the query `in keeping with`. After searching inside a TM based on the English and French sentences pairs from the Canadian Hansards, TransSearch returned those pairs in which the query occurs. The English sentences are displayed along with their French translation. Whereas the English words matching the query can be easily highlighted (here in bold), their corresponding French translation is much harder to identify.

Currently, this identification is left to the user, and has to be repeated over many sentence pairs in order to discover the set of available query translations in the TM. Indeed, sentences are not displayed in an order related to the diversity of translations occurring in the TM, but in the reverse chronological order of dates of the documents they were extracted from. Although professional translators are usually quick to spot the appropriate translation in context, this task remains very time-consuming. Relying on the user to find the corresponding translation not only restricts the benefit and the audience of TransSearch, but it also prevents its use in other natural language processing applications.

In fact this highlights a severe limitation shared by many current CAT tools: the fact they mainly rely on sentence-level matching to exploit their TM. This major drawback can be addressed with word alignment techniques, which are commonly used in statistical machine translation (SMT). A recent attempt has been made by Callison-Burch et al. (2005) who proposed the Linear B^2 system where translations of a user's query are presented in context. Unfortunately, this system's evaluation is very limited, as discussed in Section 6. In this paper, we take inspiration from this work while attempting to explore one step further in that direction.

The term *translation spotting*, coined by Véronis and Langlais (2000) and relabeled here as *transpotting*, is defined as the task of identifying the target language word-tokens that correspond to a given source language query in a pair of sentences known to be mutual translations. We call *transpot* the target word-tokens automatically associated with a query in a given pair of sentences. For instance in Figure 2, `conforme à` and `respecte` are 2 out of 213 distinct transpots displayed to the user for the query `in keeping with`.

The first contribution of this paper is to propose several transpotting methods. Further, once translations have been identified using these methods, it becomes possible to merge those translations that are identical or at least very close (if they differ only by a plural inflection for example). This allows TransSearch to display each translation

[2] `http://linearb.co.uk/`

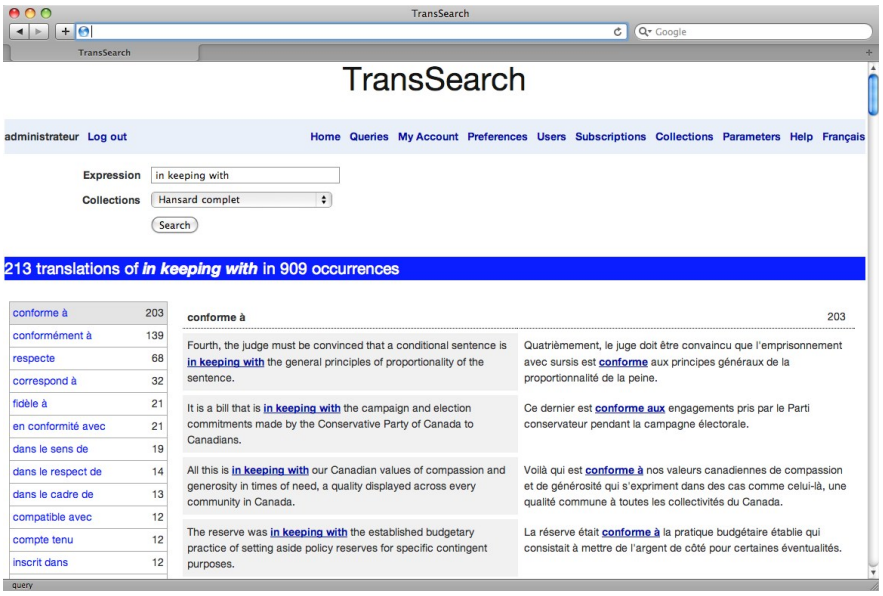

Figure 2. Results for the query `in keeping with` with the new version of TransSearch. This version displays on the left hand side the whole range of translations found in the TM. For the first suggested translation, `conforme à`, four out of the 203 sentence pairs containing a variant of this translation (see the merging process described in Section 3.2) are displayed in context. The highlighted translations are HTML links to their occurrence in the original Hansards session.

to the user only once, while keeping the possibility of consulting their contexts of occurrence. The list of translations is sorted in decreasing order of their frequency in the TM, according to the hypothesis that the most frequent translations are the most likely ones. They are displayed in a web interface, loosely inspired by the one pioneered by Linear B. CSS and Javascript are used to selectively display translation pairs. TransSearch has now become a translation finder as shown in Figure 2. The list of query translations found in the TM is presented to the user who can browse the associated pairs of sentences by clicking on each translation. Both the query and its translation are highlighted in all pairs of sentences.

The second contribution of this paper is to propose an evaluation framework for this relatively *unusual* work in MT research. We rely on a large reference corpus to conduct a series of experiments. According to the goals of the system, we define two tasks, for which we propose different metrics, to compare several translation spotting methods and post-processings. Finally, a human evaluation shows the relevance of the system's results.

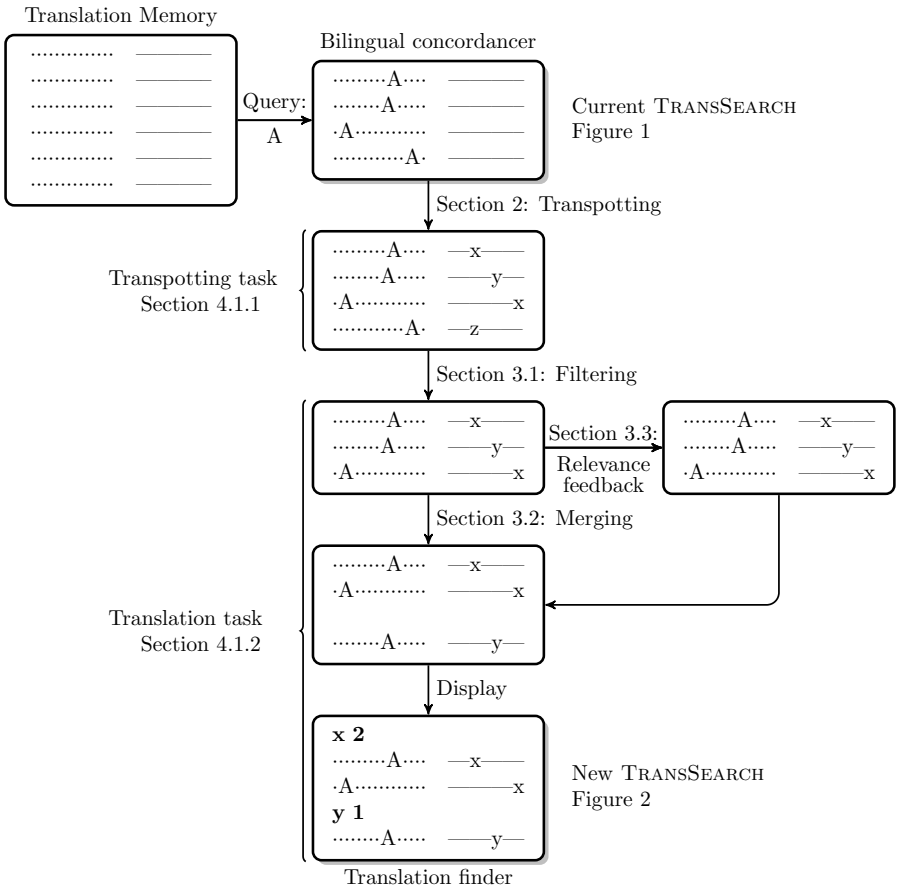

Figure 3. Processing steps for transforming the bilingual concordancer TransSearch into a translation finder. Each rounded rectangle depicts a set of translation pairs (sentences are stylized as dots for source and lines for target) returned from a query in the TM. The query is indicated by a capital letter in the source and the corresponding transpot, by a lower-case letter. The top rectangle in the center column, in which the query is highlighted in the source sentence only, corresponds to the output of the current TransSearch (see Figure 1). The steps below it illustrate the processes described in this article. The first step, transpotting, identifies transpots in the target sentences, before bad transpots are filtered. This set of transpots can optionally be refined with relevance feedback information. The resulting transpots are merged so that close variants of a same translation form a single group. The groups displayed in the lowest rectangle represent the final result (see Figure 2).

Figure 3 acts as a roadmap for the paper since it summarizes the steps we developed to identify and display query translations. Each step is associated with a corresponding section in the paper. Section 2 describes methods for spotting translations in context, and Section 3 introduces post-processing steps to enhance and display the results.

Then, the evaluation process is presented: Section 4 describes the evaluation tasks we defined, the corpora, as well as the training and tuning setups; Section 5 reports the results of our experiments. Section 6 compares this work with related work, while Section 7 concludes and explores further perspectives.

2. Transpotting

The first idea that comes to mind for transpotting a query is to use a bilingual dictionary, in which translations for multi-word expressions are often difficult or awkward to find. To overcome these limitations, we can rely on word alignment models which are commonly used in SMT. As suggested by Simard (2003b), IBM models can be used to compute the maximum a posteriori alignment, or Viterbi alignment, of a sentence pair. Then, the target words aligned to the query can be considered as the transpot.

Since IBM models are the bread-and-butter of today's SMT systems, we first describe a baseline model making use of IBM model 2. As we discuss in Section 6.1, several alternatives to these models have been recently developed but none of them emerged as a new standard for word alignment. Further, many of them rely on external resources such as manually annotated bitexts or syntactic parsers. This conflicts with two important capabilities of the current TransSearch version: the consideration of several language pairs, and the indexing of corpora in different domains. The implementation of these capabilities in the new TransSearch version prevents the use of models that rely on external resources. Therefore, we decided to focus our study on two widespread models: Hidden Markov Models (HMMs) and phrase-based models.

Further, in order to enhance the classical use of IBM models, Simard (2003b) suggests taking into account a contiguity constraint for transpotting. In keeping with this work, we also analyze the use of this constraint and present two algorithms based on IBM model 2 and HMM.

2.1. Transpotting using IBM models

Formally, given a source sentence $S = s_1...s_n$ and a target sentence $T = t_1...t_m$ in translation relation, an IBM-style alignment $a = a_1...a_m$ connects each target token to a source one ($a_j \in [1, n]$) or to the so-called NULL token which accounts for untranslated target tokens, and which is arbitrarily set to the source position 0 ($a_j = 0$).

Several word alignment models are introduced and discussed in Brown et al. (1993). They differ by the expression of the joint probability of a target sentence and its alignment, given the source sentence.

The IBM model 2 is expressed by:

$$p(t_1^m, a_1^m|s_1^n) = p(m|n) \prod_{j=1}^{m} p(t_j|s_{a_j}) \times p(a_j|j,m,n) \quad (1)$$

where $p(m|n)$ is the length distribution, the first term inside the product is the transfer or lexical distribution and the second one is the alignment distribution.

Given this decomposition of the joint probability, the so-called Viterbi algorithm finds the alignment $\hat{a}$ maximizing the quantity $p(a_1^m|t_1^m, s_1^n)$:

$$\hat{a}_j = \arg \max_{i \in [0,n]} [p(t_j|s_i) \times p(i|j,m,n)] \quad (2)$$

This computation can be done efficiently in $O(mn)$. Then the transpot of the query is obtained by returning the words t_j aligned with the words of the query according to $\hat{a}$. It is well known that IBM model 2 is a weak translation model. An easy way to enhance this transpotting method consists in using a more accurate lexical distribution. To this end, we use the transfer distribution resulting from the training of an IBM model 4, which allows us to enhance significantly the transpotting method, while it does not induce higher cost at runtime. This method is named `IBM2` in the remainder of the paper. The IBM model 4 lexical distribution is also used for all transpotting methods described below, except the method of Section 2.2.

A natural way to improve this baseline transpotting method is to make use of richer word alignment models. The introduction of a first-order alignment dependency to the IBM model 2 gives the HMM alignment model first proposed by Vogel et al. (1996). It is expressed by:

$$p(t_1^m, a_1^m|s_1^n) = \prod_{j=1}^{m} p(t_j|s_{a_j}) \times p(a_j|a_{j-1}, n) \quad (3)$$

The computation of the Viterbi alignment based on HMM can be done in $O(mn^2)$ with dynamic programming by maximizing the quantity $Q(i,m)$ computed using the following recurrence:

$$\begin{aligned} Q(i,1) &= p(t_1|s_i) \\ Q(i,j) &= p(t_j|s_i) \max_{i' \in [1,n]} [p(i|i',n) \times Q(i', j-1)] \end{aligned} \quad (4)$$

with $i \in [1,n], j \in [2,m]$. In the following, this transpotting algorithm is named `HMM`.

Since IBM models are asymmetrical, a common trend for improving their predictions consists in combining models trained in both translation directions (English–French and French–English). Several operators

can be used, such as intersection, union or grow-diag-final, as defined in Och and Ney (2003). We tested the use of symmetrized HMMs for transpotting with intersection and union operators; these transpotting methods are called `HMM-bi-inter` and `HMM-bi-union` respectively in the following. Those algorithms have a complexity in $O(mn^2)$. Because the transpotting method described in Section 2.2 relies on the combination of HMM models with the grow-diag-final operator, we did not test further this operator with the models described in this section.

2.2. Phrase-based transpotting

As proposed by Callison-Burch et al. (2005), phrase-based models can also be used by a translation search engine. The Moses toolkit produces an SMT system relying on a phrase table that combines HMM alignments using the grow-diag-final operator (Koehn et al., 2007).

From the SMT system produced by Moses, we are only interested in the phrase table that records the word alignment knowledge. In this work, we implemented a simple strategy in which the transpots returned by the system are those present in the phrase table for the query and which belong to the target sentence. The resulting set can be sorted using some combination of the scores associated with each phrase pair. With Moses, these scores are phrase translation probabilities and lexical weightings in both translation directions. The combination we retained in our study has been tuned on development data, as described in Section 4.4.

A suffix array structure offers a convenient way of indexing the huge phrase table resulting from our large training corpus of 8.3 million sentence pairs (Callison-Burch et al., 2005). In the following, we name this approach `PBM`.

2.3. Transpotting using IBM models and a contiguity constraint

Whereas selecting all the target tokens aligned with the query is a straightforward transpotting method, this strategy is error prone and a better transpotting algorithm deserves to be considered. Figure 4 illustrates a common error that appears when using `IBM2` for transpotting. In this example, the identified transpot for the query `in keeping with` is a non-contiguous phrase composed of `mesure` and `conforme à`. Although it may be necessary to choose a non-contiguous phrase, contiguous tokens in the source language sentence tend to be aligned to contiguous tokens in the target language. As mentioned in Simard (2003b), this suggests that it is relevant to integrate a contiguity constraint inside the transpotting algorithm. This idea, which shares some similarity

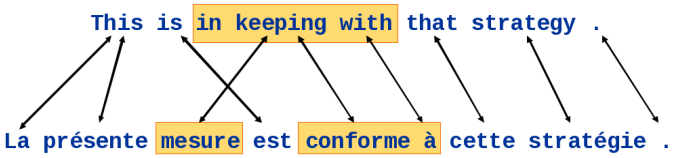

Figure 4. Example of word alignment generated by an IBM model 2 that leads to an erroneous transpot for the query `in keeping with`.

with the phrase extraction technique described in Vogel (2005), can be expressed as follows.

For each target token pair $(j_1, j_2) \in [1, m] \times [1, m], j_1 < j_2$, two Viterbi alignments are computed: one between the phrase $t_{j_1}^{j_2}$ and the query $s_{i_1}^{i_2}$, and one between the remaining material in the two sentences $\bar{s}_{i_1}^{i_2} \equiv s_1^{i_1-1} s_{i_2+1}^{n}$ and $\bar{t}_{j_1}^{j_2} \equiv t_1^{j_1-1} t_{j_2+1}^{m}$. This method finds the translation of the query according to:

$$\hat{t}_{\hat{j}_1}^{\hat{j}_2} = \underset{(j_1,j_2)}{\operatorname{argmax}} \left[p(a_{j_1}^{j_2} | s_{i_1}^{i_2}, t_{j_1}^{j_2}) \times p(\bar{a}_{j_1}^{j_2} | \bar{s}_{i_1}^{i_2}, \bar{t}_{j_1}^{j_2}) \right] \tag{5}$$

Whereas the first term of this equation ensures the contiguity constraint by forcing the query $s_{i_1}^{i_2}$ to be aligned to a contiguous segment $t_{j_1}^{j_2}$, the second term forces each token of $\bar{t}_{j_1}^{j_2}$ to be aligned to a source token outside the query.

2.3.1. *With IBM model 2*

Implementing the contiguity constraint for IBM model 2 can be naively achieved by computing m^2 Viterbi alignments, one for each pair (j_1, j_2), which corresponds to a complexity in $O(m^3 n)$. Fortunately, because the source query is given, it is possible to compute more efficiently this solution by introducing three tables corresponding to three states of the alignment procedure: before the transpot, inside the transpot and after the transpot. These tables are computed by dynamic programming:

$$\begin{aligned}
Q_{\text{before}}(1) &= \max_{i \notin [i_1,i_2]} P(t_1|s_i) \\
Q_{\text{inside}}(1) &= \max_{i \in [i_1,i_2] \cup \{0\}} P(t_1|s_i) \\
Q_{\text{after}}(1) &= 0 \\
Q_{\text{before}}(j) &= \max_{i \notin [i_1,i_2]} \{ Q_{\text{before}}(j-1) p(t_j|s_i) p(i|j,m,n) \} \\
Q_{\text{inside}}(j) &= \max_{i \in [i_1,i_2] \cup \{0\}} \{ \max(Q_{\text{before}}(j-1), Q_{\text{inside}}(j-1)) \times \\
&\quad p(t_j|s_i) p(i|j,m,n) \} \\
Q_{\text{after}}(j) &= \max_{i \notin [i_1,i_2]} \{ \max(Q_{\text{inside}}(j-1), Q_{\text{after}}(j-1)) \times \\
&\quad p(t_j|s_i) p(i|j,m,n) \}
\end{aligned} \tag{6}$$

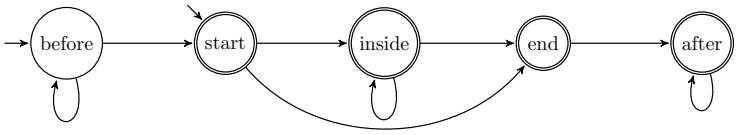

Figure 5. Automaton with the 5 allowed states for the transpotting method based on HMM and the contiguity constraint: before the transpot ($j < j_1$), start of the transpot ($j = j_1$), inside the transpot ($j_1 < j < j_2$), end of the transpot ($j = j_2$) and after the transpot ($j > j_2$).

with $i \in [0, n], j \in [2, m]$. Let us note that $Q_{\text{inside}}(j)$ and $Q_{\text{after}}(j)$ require a second maximization operator to choose the best scores obtained for the token $(j-1)$ in the two previous allowed states. The tables are computed efficiently in $O(mn)$, while the best transpot corresponds to $\max(Q_{\text{inside}}(m), Q_{\text{after}}(m))$. This method will be referred to as `C-IBM2` in the remainder of the article.

2.3.2. *With HMM*

The contiguity constraint can rely on HMMs as well. Whereas the naive implementation gives a complexity in $O(m^3n^2)$, a careful implementation allows a computation in $O(mn^2)$. This exploits the existence of five states defined according to the current value of j with respect to hypothesized j_1 and j_2 (see Figure 5). Thus, the table $Q(i, j)$, required to compute a Viterbi alignment based on an HMM, can be duplicated for each of the five states and computed according to the principle described in Section 2.3.1.

For example, let us consider the table $Q_{\text{end}}(i, j)$ of the state reached at the end of the transpot. It is built from the tables $Q_{\text{start}}(i, j)$ and $Q_{\text{inside}}(i, j)$, associated with the two previous states allowed in the automaton. Its value are computed as follows:

$$\begin{aligned} Q_{\text{end}}(i, 1) &= 0 \\ Q_{\text{end}}(i, j) &= \begin{cases} p(t_j|s_i) \max_{i' \in [i_1, i_2]} \{p(i|i', n)\eta(i', j-1)\} & \text{if } i \in [i_1, i_2] \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{7}$$

with $\eta(i', j-1) = \max(Q_{\text{start}}(i', j-1), Q_{\text{inside}}(i', j-1))$, $i \in [1, n], j \in [2, m]$. The best transpot is finally chosen as the one that maximizes $Q(i, m)$ among the four accepting states. We name this method `C-HMM`.

2.3.3. *In both directions*
The transpotting methods incorporating a contiguity constraint described in Section 2.3 can be improved by combining models in both translation directions. For this, the transfer probabilities of the French–English and English–French models are combined, using some combination function tuned on the development data as described in Section 4.4. This results in a bidirectional transpotting method we call `C-HMM-bi` for HMM, with a complexity remaining in $O(mn^2)$.

Among all transpotting methods we described in Section 2 and according to the experimental assessment presented in Section 5, `C-HMM-bi` obtains the best performance in the context of our application.

3. Post-processing

Queries that occur frequently in the TM receive numerous translations using the transpotting methods described above. For example, Table I illustrates the many transpots returned by `C-IBM2` for the query `on behalf of`. Some transpots (those marked with a star) are clearly wrong (*e.g.* `de`), others (in italics) are only partially correct (*e.g.* `part de`). Also it appears that many transpots are very similar (*e.g.* `au nom de` and `au nom du`, where `du` is the contracted form of `de le`).

Since we want to present to the user a list of translations corresponding to the query, strategies must be devised for dealing with some errors resulting from the transpotting phase. We estimate that a user will focus on the 10 first translations presented, so we want to provide as many correct and diversified translations as possible at the top of the result page.

The number of translations to discard can be significant: using the transpotting algorithms described above, we observed that roughly between 20 % and 40 % of suggested translations are erroneous. Most of them have few occurrences and appear at the end of the list, nevertheless even the top results can contain a few errors. For example, in Table I the fourth translation `de` is incorrect while it was found 136 times.

We investigated three avenues to enhance the results of the transpotting phase: filtering erroneous transpots (Section 3.1), merging variants of the same canonical translation (Section 3.2), and the adaptation of pseudo-relevance feedback for transpotting (Section 3.3).

Table I. Subset of the 824 different transpots retrieved for the query `on behalf of`. Those marked with a star are erroneous, those in italics are only partially correct.

Transpot	Frequency
au nom de	1424
au nom du	763
au nom des	683
de⋆	136
...	
dans l'intérêt des	15
de la part de	13
dans⋆	13
part de	13
...	
pour l'ensemble de	1
parler pour	1
loin que⋆	1
le bien⋆	1

3.1. Filtering bad transpots

A simple way to detect and filter bad transpots is to rely on supervised learning. To this end, we analyzed a set of queries and their transpots, as computed by the `C-IBM2` transpotting method (our best transpotting algorithm available at the moment), and manually annotated the transpots as "good" or "bad". This corpus (see Section 4.2.3) was used to train a number of classifiers designed to distinguish good transpots from bad ones.

We experimented with several popular classifiers:[3] a *support vector machine* (SVM), commonly used in supervised learning (Cristianini and Shawe-Taylor, 2000); a *multi-layer perceptron* with one level of hidden neurons (Bishop, 1995); *AdaBoost* using a one level decision tree as weak classifier (Freund and Schapire, 1996); a *bagging* algorithm using a decision tree as base classifier (Breiman, 1996); and a *random forest* algorithm (Breiman, 2001). Besides, these five classifiers were combined through a *linear combination* by averaging over their posterior class probabilities (Kittler et al., 1998).

We computed three groups of features for each example, that is, each query/transpot pair (q, t). The first group is made up of features

[3] Available in the Weka framework at `http://www.cs.waikato.ac.nz/ml/weka`

related to the size (counted in words) of q and t, with the intuition that they should be related. The second group gathers various alignment scores computed with word alignment models (Viterbi scores using IBM models 1 & 2 in both directions; min, max and average likelihood scores inside Viterbi alignments, etc.). The last group gathers clues that are more linguistically flavored, among which the ratio of grammatical words (like `for`, `although`, `the`) in q and t, or the number of prepositions and articles. In total, each example is represented by 45 numerical features.

3.2. Merging variants

Once erroneous transpots are filtered out, there usually remain many translations for a given query. For instance, for the query `on behalf of`, our best classifier identified 213 bad transpots among 824 candidates. Some of the remaining transpots are very similar and are not interesting for the user (see Table I). This phenomenon is particularly acute in French with its numerous conjugated forms of verbs. Still many transpots differ only by punctuation marks or by a few grammatical words.

We propose merging two translations which are different inflectional forms of the same sequence of canonical words. For instance, `au nom du` and `au nom des` from Table I will be considered as similar, since `du` and `des` are contractions of `de + le` and `de + les` respectively, where `le` and `les` are definite articles. Furthermore, we noticed that displaying translations that differ only by a few grammatical words or punctuation marks, like `de la part de` and `part de` is often redundant for the user, so these are combined as well.

This clustering process must be fast in order to be used online. For this, each translation is first associated with a key: the grammatical words are ignored (in this case: determiners, prepositions, conjunctions, pronouns and auxiliary verbs); then each word of the translation is replaced by all the lemmas which can generate it, according to a lemma dictionary. For translations containing only grammatical words, the key consists of the words themselves. This avoids clustering all grammatical word sequences into a single group, hence `de` and `dans` remain in different clusters.

This first step does not permit disambiguation of translations according to their lemmas. For instance, `part de` which can be the verb `partir` (in English, `leave`) or the noun `part` (in English, `part`) is associated with the key {`partir`, `part`}. To merge this translation with another one that does not exhibit a part-of-speech ambiguity (e.g. `partira` (in English, `will leave`) which is associated with the key {`partir`}),

Algorithm 1:

Data: S_0: set of translations
Result: h_2: hash map whose values are sets of similar translation
initialize hash map h_1 ; `//step 1: search lemmas`
foreach $t \in S_0$ **do**
  $u \leftarrow$ remove grammatical words of t
  if $u = \epsilon$ **then** $key \leftarrow lemmas(t)$
  else $key \leftarrow lemmas(u)$
  add t to $h_1[key]$
end
initialize hash map h_2 ; `//step 2: disambiguate lemmas`
foreach k_1 *of the keys of* h_1 *sorted in decreasing order of transpotting frequency* **do**
  $k \leftarrow k_1$
  $S \leftarrow h_1[k_1]$
  foreach k_2 *of the keys of* h_2 *sorted in decreasing order of transpotting frequency* **do**
    if $k_1 \cap k_2 \neq \emptyset$ **then**
      $k \leftarrow k_1 \cap k_2$
      $S \leftarrow h_1[k_1] \cup h_2[k_2]$
      remove k_2 from h_2
      break
    end
  end
  $h_2[k] \leftarrow S$
end

Figure 6. Variant merging process where the function *lemmas* gives the set of all the possible lemmatized sequences of a translation according to a dictionary.

these keys are, in a second step, disambiguated according to a greedy process that computes the intersections of the sets of lemmatized sequences. In order to avoid disambiguation errors, frequent transpots are considered first since they are usually correct. Figure 6 shows the algorithm of this variant merging process.

3.3. Pseudo-relevance feedback

We studied a third post-processing stage on top of the two aforementioned ones. We investigated exploiting the fact that many sentence pairs containing the query share the same set of translations which may be helpful to refine transpotting results. The transpotting methods described in Section 2 align each sentence pair individually — as it is done in SMT. It is therefore interesting to know whether the alignment

Table II. Subset of the 226 different transpots retrieved for the query `way of life` by the transpotting algorithm during the first stage.

Transpot	Frequency
mode de vie	898
mode de vie des	42
façon de vivre	36
style de vie	32
niveau de vie	8
manière de vivre	5
de vie	5
mode de vie de	5
...	
qualité de vie	3
façon d'être	3
...	
vie rurale	2
réalité dans nos	1
à la société	1
au cœur du mode de vie	1
...	

of a given query can be enhanced by taking into account the alignment of this query in other sentence pairs containing it.

In this section, we present two methods that release the usual independence assumption between sentence pairs. For this, the information provided by the translations found during a first transpotting stage is used to refine the results during a second transpotting stage. These methods are similar in principle to the *relevance feedback* concept developed in the information retrieval (IR) domain (Rocchio, 1971).

Usual relevance feedback techniques rely on human judgments for identifying relevant documents returned during a first retrieval stage; this information is used for improving a second stage. A variant of this method, known as pseudo-relevance feedback, does not require user annotation but assumes that the top ranked documents returned during the first stage are relevant (Croft and Harper, 1979). In our case, a first transpotting phase is carried out in which the most frequent transpots are considered as the most relevant. This information is then used to improve a second transpotting stage.

We noticed that frequent transpots tend to be good translations of a query. This is illustrated in Table II for the query `way of life`. The correct translation `mode de vie` clearly occurs more frequently than the other transpots. The next candidates are also relevant translations, such as `façon de vivre` or `style de vie`. At the end of the list, many transpots, especially those occurring only once, are incorrect (e.g. `réalité dans nos`) or correspond to variants of the most frequent translation (e.g. `au cœur du mode de vie`). We now present two methods we designed according to this principle.

3.3.1. *Procedural relevance feedback*
Based on the observation that frequent transpots are likely to be good ones, the set of the most frequent transpots is first built for each query, then rare transpots are replaced by an element of this set. Each rare transpot found in a given sentence is replaced by the most frequent transpot occurring in the sentence. If no frequent transpot occurs in the sentence, the transpot is left unchanged. We call this method procedural relevance feedback, or `PRF` for short. The decision to consider a transpot as rare is based on two parameters α and β, which respectively set an absolute and a relative threshold. For example, the values $\alpha = 5$ and $\beta = 0.02$ consider as rare transpots those occurring less than 5 times and in less than 2 % of the retrieved sentence pairs. These two parameters are optimized on a development corpus (see Section 4.2.2).

3.3.2. *Statistical relevance feedback*
The previous relevance feedback method has two drawbacks: it can only replace a transpot with a more frequent one, and it only uses the results provided by the transpotting method from a static word alignment model. We propose a statistical relevance feedback method, named `SRF`, which attempts to improve the statistical word aligner. For each query, a *local* statistical transfer model is computed using the transpots found during the first stage, with the hope of improving transpotting during the second stage. In order to do so, we build for each query a parallel corpus made of the query and all the transpots found during the first stage. We assume that this short parallel corpus contains information which is more specific to the translation of the query than the very large training corpus used to build the main transfer model. Let us note that we do not compute a local alignment model: since the parallel corpus for each query is small with respect to the bitext used to train the global model, a local alignment model would only damage the global alignment model.

The specific corpus is used to compute the probabilities $p_{\text{loc}}(t_j|s_i)$ of a local transfer model which are linearly interpolated with the prob-

abilities $p_{\text{glob}}(t_j|s_i)$ of the global transfer model initially used by the transpotting algorithm. Because the local bitext is very short, training the local model is very fast. This idea shares commonalities with the cache model used in language modeling (Kuhn and De Mori, 1990). Since the specific corpus only provides information about the use of the words of the query, the modifications of the transfer model are limited to those words. Therefore, the new transfer distribution used during the second transpotting stage becomes:

$$p(t_j|s_i) = \begin{cases} \lambda p_{\text{glob}}(t_j|s_i) + (1-\lambda) p_{\text{loc}}(t_j|s_i) & \text{if } s_i \in q \\ p_{\text{glob}}(t_j|s_i) & \text{otherwise} \end{cases} \quad (8)$$

where λ is a coefficient optimized on a development corpus.

4. Experimental setup

We now describe the metrics we designed and the data sets we gathered for the evaluation. Then we present the training and the tuning procedures used for the transpotting algorithms.

4.1. Metrics

The new TransSearch prototype achieves two related tasks that deserve their own evaluation: the transpotting and the translation tasks. On one hand, the transpotting task corresponds to the use of TransSearch as a bilingual concordancer: as shown in Figure 2, for each sentence pair the application highlights the words that form the transpot of the query. On the other hand, the translation task corresponds to the use of TransSearch as a translation finder where the system presents the transpots corresponding to a user's query.

Although these two tasks are strongly related, their outputs are different: for the transpotting task subsequences of sentence pairs are highlighted, while for the translation task groups of translations are produced. Further, the transpotting task only depends on the quality of the transpotting algorithms, whereas the translation task relies also on the post-processing. We now describe the metrics we designed to evaluate each task.

4.1.1. *Transpotting task*

The transpotting evaluation concerns the ability of an algorithm to identify the reference transpot in a target sentence. This task can be evaluated after the transpotting stage, and also after the pseudo-relevance feedback stage which aims at correcting erroneous transpots in the sentence pairs retrieved for a user query.

Following the previous work of Simard (2003b), the relevance of a transpot r for a given sentence pair (s,t) can be measured in terms of precision and recall when comparing r with the reference transpot $\hat{r}$ obtained from a bilingual lexicon (see Section 4.2.2). Scores are computed as follows:

$$\begin{aligned} \text{recall}(s,t) &= |LCS(r,\hat{r})|/|\hat{r}| \\ \text{precision}(s,t) &= |LCS(r,\hat{r})|/|r| \end{aligned} \tag{9}$$

where $LCS(r,\hat{r})$ returns the longest common contiguous subsequence of tokens shared by r and $\hat{r}$, and $|r|$ denotes the length of string r. Thus, transpots that are only partially correct are given some credit depending on the length of overlap with the reference. This reflects the capacity of a transpotting algorithm to identify where the transpot is located in the target sentence. This capacity is very useful for the user: it avoids the need to search for the translation in the whole target sentence, even if an algorithm identifies only a grammatical word of the reference transpot.

The previous scores are determined at the level of sentence pairs and must therefore be averaged to get recall and precision ratios at the corpus level. To do so, the scores are first averaged for each query/reference transpot pair, then averaged over all the pairs of query/reference transpots for each query, and finally averaged over all the queries. These three levels of average reduce the evaluation bias toward queries associated with numerous sentence pairs with respect to others having only a few occurrences in the TM. They also prevent the scores from giving more frequent reference translations a too important weight with respect to rare ones in the TM. Finally, precision and recall can be combined into an F-measure score in the usual way:

$$\text{F-measure} = \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}} \tag{10}$$

4.1.2. *Translation task*

The translation evaluation reflects the ability of an algorithm to find the different translations of a given query in the retrieved pairs of sentences. This task is essential for TransSearch since the results must be displayed within a limited amount of space, which requires them to be both correct and diversified. This task can be evaluated after all the processing stages, because they either find transpots that can be directly turned into lists of transpots (transpotting and pseudo-relevance feedback stages) or are specifically designed to improve the quality of the displayed translations (filtering and merging stages).

The evaluation of this task is based on the fact that returning a list of translations for a submitted query is similar to what happens in

a classical IR system, which retrieves a list of documents for a given query. Like in the IR domain, we expect the translations returned for a query to be ordered by relevance, with the most interesting at the beginning of the list.

The *mean average precision* (MAP) measure, now commonly used in IR (Manning et al., 2008), provides a single-number measure of quality across recall levels which gives a higher weight to top levels. In our case, MAP_k is established at the rank k by comparing the top k translations with the ones available for each query in a reference lexicon (see Section 4.2.2). The MAP_k score averages the precision scores obtained at each rank, up to k, that corresponds to a relevant translation. Formally, let x_j denote the j^{th} translation returned in the list, and $\mathcal{Y}$ the set of reference translations. Then, MAP_k is defined as the average of the following score computed for each query:

$$\frac{1}{|\mathcal{Y}|}\sum_{i=1}^{k}\mathbb{1}_{x_i\in\mathcal{Y}}\frac{\sum_{j=1}^{i}\mathbb{1}_{x_j\in\mathcal{Y}}}{i} \tag{11}$$

where $\mathbb{1}_{\text{p}}$ equals 1 when p is true and 0 otherwise.

The MAP only manages two levels of relevance for translations: right or wrong, while measures handling a degree of relevance can also be very useful. Indeed, among relevant translations, some can be more interesting than others: a rare translation which occurs in the TM but not in a bilingual dictionary can be considered as more interesting than a common translation. In order to handle the relevance of translations, we rely on the *Q-measure* which associates documents with different relevance scores ranging from 0 to 1 (Sakai, 2004). We apply this metric to our context by considering lists of translations as lists of documents. Formally, the Q-measure associates each translation x with a relevance score $\text{rel}(x) \in [0,1]$. $\mathcal{Y}$ is defined as the set of reference translations y annotated with a relevance score $\text{rel}(y) \in]0,1]$. An optimal list of translations is built by ordering $\mathcal{Y}$ in decreasing order of relevance; the j^{th} element being denoted by y_j. The Q-measure at rank k is defined as the average of the following score computed for each query:

$$\frac{1}{|\mathcal{Y}|}\sum_{i=1}^{k}\mathbb{1}_{x_i\in\mathcal{Y}}\frac{\sum_{j=1}^{i}\mathbb{1}_{x_j\in\mathcal{Y}}.(\text{rel}(x_j)+1)}{i+\sum_{j=1}^{i}\text{rel}(y_j)} \tag{12}$$

The Q-measure corrects the incapacity of the weighted MAP measure (Kando et al., 2001) to rank two document lists which differ only by some document beyond the $|\mathcal{Y}|^{\text{th}}$ position in the list. To this end, the constant 1 is added in the numerator, and i in the denominator. Otherwise, ratios would remain constant beyond the $|\mathcal{Y}|^{\text{th}}$ position.

In Section 5, we associate a relevance score to each transpot according to its frequency in the TM; this allows the comparison of different transpotting methods through the Q-measure.

4.2. Corpora

4.2.1. *Translation memory*

The largest TM used in TransSearch comes from the Canadian Hansards, a collection of the official proceedings of the Canadian Parliament. For our experiments, we used an in-house sentence aligner (Langlais, 1997) to align 8.3 million French–English sentence pairs extracted from the 1986–2007 period of the Hansards. Then, this bitext was indexed with Lucene[4] to form our TM.

4.2.2. *Evaluation corpus*

To study the behavior of our methods on "real" queries, we extracted from the log file of the current TransSearch the 5,000 most frequent English queries that were submitted to the system during the 2001–2007 period. The manual evaluation of the transpots suggested by a transpotting method is a long and often difficult task. To avoid this inspection, we built automatically a large reference corpus using a bilingual phrase lexicon we collected over various projects. Among the 5,000 queries that have an entry in this lexicon, 284 queries were used for development purposes (dev) and 2,074 were kept apart for testing our methods (test). For each query, up to 5,000 sentence pairs were retrieved from the TM.

The goal of the transpotting task is to highlight the words of the target sentence that translate the query. This requires a reference corpus composed of sentence pairs annotated with a reference transpot. Among the 5,000 pairs of sentences retrieved for each query of dev and test, we kept only those whose source part contains the query and target part contains one of its translations in the bilingual lexicon. This resulted in a set containing 180,000 pairs of sentences for dev and 1.4 million pairs of sentences for test.

On the other hand, the translation task requires a list of translations produced by a transpotting method to be compared with a list of reference translations. The reference translations are those occurring in the bilingual lexicon, which contains an average of 3.6 translations per query for dev and 3.9 for test. We are aware that the limited amount of translations might bias our evaluation. Nevertheless, this bias is uniform for all the transpotting methods.

[4] `http://lucene.apache.org`

4.2.3. *Classifier training corpus*
In order to train the classifiers described in Section 3.1, four human annotators were asked to identify bad transpots among those proposed by one of our transpotting algorithms. We developed an ad-hoc web-based interface that displayed a query with its corresponding transpots. The annotator indicated whether each transpot was appropriate or not. Annotating the query/transpot pairs without their contexts of occurrence allows a relatively fast annotation process, around 40 seconds per query, but leaves some difficult cases to annotate (more details are given in Section 5.5). For instance, for the query `in keeping with`, `conforme à` is straightforward to annotate, but others such as `dans le sens de` or `tenir compte de` gave annotators a harder time since both are contextual translations which are valid in some specific contexts.

We ended up with a set of 531 queries that have an average of 22.9 transpots each, for a total of 12,144 annotated examples. We computed the Fleiss' inter-annotator agreement (Fleiss et al., 2003) and observed a 0.76 kappa score, which suggests a high degree of agreement (Landis and Koch, 1977). When annotations differ, the reference finally considered is randomly selected among the judgments done by the annotators.

4.3. Model training setups

All the transpotting methods described in Section 2 use IBM models. To obtain them, we ran Giza++ (Och and Ney, 2003) on our 8.3 million sentence pairs TM with the default parameter set. We used the following sequence of models with five training iterations for each: 1, 2, HMM, 3 and 4. This provided us the transfer tables corresponding to the IBM model 4 which we used for all transpotting methods in place of their usual transfer table (except for `PBM` which does not use a lexical transfer table). This enhances the weakest transpotting methods, such as `IBM2`, making them stronger baselines.

To build the phrase table required by the `PBM` transpotting method, we applied Moses on the TM with the default parameter set and the sequence of models described above.

4.4. Transpotting algorithm tuning

The two transpotting methods `C-HMM-bi` and `PBM` have to be tuned. For this, we made use of the dev corpus described in Section 4.2.2 and optimized these methods according to the F-measure score for the transpotting task described in Section 4.1.1.

The `C-HMM-bi` transpotting method relies on a function that combines transfer probabilities of models trained in both directions. Several

Table III. The results (in %) of the various transpotting algorithms for the transpotting task on TEST.

	Recall	Precision	F-measure
IBM2	76.7	65.8	70.8
HMM-bi-inter	68.2	77.4	72.5
HMM	80.3	69.1	74.3
C-IBM2	77.6	74.4	76.0
HMM-bi-union	**85.0**	70.4	77.0
C-HMM	80.4	75.6	77.9
PBM	81.7	77.4	79.5
C-HMM-bi	80.9	**78.3**	**79.6**

combinations were tried and the one weighting the French given English model three times as much as the other model obtained the best results.

The PBM method requires a score associated with each phrase pair in order to sort them. For each phrase pair, MOSES produces five scores which can be combined to obtain the required single score. We tried several score combinations and obtained the best results using the same weights as C-HMM-bi for the French given English scores (phrase translation probability and lexical weighting) and the English given French scores, while the last (constant) score is ignored.

5. Experiments

In this section, we first compare the transpotting algorithms described in Section 2 and analyze the impact of the post-processing presented in Section 3. Finally, we report the results of a human evaluation for three variants of our prototype and provide guidelines for integrating the methods in the production version of TRANSSEARCH.

5.1. EVALUATION OF THE TRANSPOTTING ALGORITHMS

Using the TEST corpus, we evaluated the transpotting algorithms described in Section 2 according to the transpotting and translation tasks defined in Section 4.1.

5.1.1. *Transpotting task*

Table III shows the results in percentage related to the transpotting task. Without much surprise and despite its use of IBM model 4 transfer

tables, IBM2 is the weakest transpotter. At the opposite end of the range, we observe that PBM and C-HMM-bi outperform the other algorithms, the latter slightly outperforming the former. A closer look at these figures shows that using C-HMM-bi induces a considerable increase of precision with respect to HMM-bi-inter and PBM. Whereas aligning all target words to the query gives a high recall, only the correct words have to be aligned to the query to obtain both high recall and precision. This means that a transpotting method has to guess the correct boundaries of the transpot with high accuracy. The results in Table III suggest that C-HMM-bi is more acute at finding these boundaries and giving a correct transpot.

The same phenomenon can be observed when comparing IBM2 to C-IBM2. The introduction of the contiguity constraint described in Section 2.3 increases recall by almost 1 absolute point and precision by almost 9 points. Further, since both HMM and C-IBM2 introduce a sort of neighborhood constraint with respect to IBM2, it is interesting to note that the improvement of F-measure induced by the contiguity constraint is almost 2 points more than the one induced by the previous word alignment constraint. This empirically validates the interest of this constraint and confirms the observations made by Simard (2003b).

5.1.2. *Translation task*

Table IV reports the results for the translation task. MAP scores are computed at ranks 5, 10, and globally for the whole list of translations. The C-HMM-bi clearly outperforms all other transpotting methods for this task for each of the three MAP scores. When comparing this method with PBM, the MAP scores are between 1.5 and 2 points higher.

The MAP metrics evaluate a list of transpots on two dimensions: the order of transpots in the list (the closer a correct translation is to the top of the list, the higher the score), and the number of retrieved correct transpots (the higher this number, the higher the score). In order to evaluate the contribution of the two dimensions, we computed Q-measure scores in which the first dimension is made explicit by the weights associated with reference translations.

As described in Section 4.1.2, Q-measure can be framed in terms of a weighted MAP. The relevance score of a translation is set to the inverse of its frequency in the TM, with the intuition that most of the systems will discover the frequent translations but only few of them will discover the less frequent ones, rendering them more interesting. Finally, as each reference translation has a relevance score, they can be ordered decreasingly in order to constitute the list of y_js defined in Section 4.1.2. This allows the computation of the Q-measure scores presented in Table IV.

Table IV. The results (in %) of the various transpotting algorithms for the translation task on TEST. MAP_k and QM_k give scores at rank k, otherwise the whole transpot list is considered

	MAP_5	MAP_{10}	MAP	QM_5	QM_{10}	QM
IBM2	33.7	36.1	38.8	23.5	27.9	46.3
HMM-bi-inter	30.4	32.8	35.3	21.0	25.6	38.8
HMM	35.2	37.7	40.3	24.7	29.5	48.0
C-IBM2	34.1	36.6	39.5	24.3	29.7	49.6
HMM-bi-union	34.8	37.3	39.7	24.3	28.9	46.1
C-HMM	35.0	37.4	40.3	24.7	29.6	50.8
PBM	33.9	36.6	39.3	24.3	29.6	45.3
C-HMM-bi	**35.5**	**38.1**	**41.1**	**25.1**	**30.3**	**51.2**

Comparing PBM and C-HMM-bi shows that the latter still obtains better scores, but two phenomena appear. On one hand, the differences between the two methods' QM_5 and QM_{10} scores are lower than those between their MAP_5 and MAP_{10} scores. On the other hand, the difference between their QM largely surpasses the one between their MAP. The first phenomenon means that the two methods tend to suggest a similar list of transpots at the beginning of the list, and have therefore the same behavior according to the first dimension described above. On the contrary, the second phenomenon clearly states that C-HMM-bi retrieves a significantly larger number of reference translations than PBM does, making C-HMM-bi more suitable for the translation task.

The queries used to constitute the TEST corpus have a wide range of occurrence frequency inside the TM. In order to study how this acts upon the quality of the results, we measured the MAP score for rare queries. Since only 9 queries of TEST occur in at most 10 sentence pairs, a larger corpus was specifically designed. From the log file of TRANSSEARCH we selected the 200 top queries that occur at most 10 times in the TM and that have an entry in our bilingual lexicon.

The results on this new corpus exhibit a significantly lower MAP_{10} score for PBM (36.6 %) with respect to C-HMM-bi using SRF (52.7 %) or not (52.3 %). This is explained by two phenomena. First, 54 out of 200 rare queries do not have an entry in the phrase table. Second, the number of translations in the phrase table is low for rare queries, which leads PBM to suggest 1.23 translations on average for the 10 sentences of each query, whereas C-HMM-bi suggests 2.77 translations on average. This observation suggests that a fruitful strategy for improving PBM on

Table V. Performance (in %) of different classifiers for identifying bad transpots.

			Bad transpots		
Classifier	Features	CCI	precision	recall	FM
Baseline: all good		61.8	0.0	0.0	0.0
Baseline: gram. ratio > 0.75		78.8	89.4	50.5	64.5
Bagging	size	73.9	74.9	46.6	57.4
	IBM	85.8	85.4	75.4	80.1
	grammatical	80.2	93.1	51.6	66.4
	all	**86.1**	86.7	74.7	**80.2**
SVM	all	84.3	82.4	75.0	78.5
Multilayer Perceptron		85.7	86.5	73.6	79.5
AdaBoost		85.6	86.4	73.8	79.6
Random Forest		85.4	84.0	76.0	79.8
Linear Combination		**86.4**	86.0	76.6	**81.0**

low frequency queries would be to consider subsequences of the query while searching in the phrase table.

Following these results, we concentrate on `C-HMM-bi` and `PBM` in the remainder of the experiments.

5.2. Filtering removes noise

5.2.1. *Bad transpot classification*

As described in Section 3.1, we trained various classifiers to identify spurious transpots from three kinds of feature sets. All these classifiers plus two challenging baselines are evaluated according to the ratio of *correctly classified instances* (CCI). Since in our application we are interested in filtering out bad transpots, precision, recall and F-measure scores related to this class are computed as well.

We report in Table V the figures computed using a 10-fold stratified cross-validation procedure. The simplest baseline (line 1) classifies all instances as good; this *useless* filter has a CCI ratio of 61.8 %. A more sensible baseline—that we engineered after we investigated the usefulness of different feature sets—classifies as bad the transpots whose ratio of grammatical words is above 0.75. It is associated with a CCI ratio of 78.8 % (line 2).

Among the five classifiers we tried, Bagging obtains the highest CCI ratio (86.1 %) and the best F-measure (80.2 %). In order to study the contribution of each feature set (size, IBM, and grammatical features), this classifier was trained using each set separately. This shows that IBM features are clearly the most significant since adding the two other feature sets only slightly improves the performance. The linear

Table VI. The results of two transpotting algorithms for the translation task on TEST after different post-processing stages.

	PBM			C-HMM-bi		
	MAP_5	MAP_{10}	MAP	MAP_5	MAP_{10}	MAP
no post-processing	33.9	36.6	39.3	35.5	38.1	41.1
filtering	34.9	37.8	40.3	35.7	38.5	41.4
filtering + merging	44.3	47.3	49.5	45.5	48.7	51.3

combination of the five classifiers gives a small increase of the CCI ratio (86.4 %) and of F-measure (81.0 %).

5.2.2. *Filtering using classifiers*
Using the best classifier according to the previous tests, *i.e.* the linear combination of five classifiers, we evaluated on the TEST corpus the benefit of discarding bad transpots. Since the filtering process is only relevant when displaying the list of translations, this evaluation was performed for the translation task only.

The first line of Table VI recalls the MAP scores computed for the `PBM` and `C-HMM-bi` methods and presented in Table IV. The second line gives the results obtained after filtering bad transpots. The comparison of these two lines shows that the three MAP scores are improved by around 1 absolute point for `PBM` and by around 0.5 points for `C-HMM-bi`. Filtering bad transpots causes the rank of good ones to decrease, which improves MAP according to the first dimension described in Section 5.1.2, *i.e.* the rank of reference translations retrieved.

When we manually check the transpot lists returned by both systems before and after filtering, we observe a significant difference at the end of the lists: a lot of bad transpots such as grammatical words or incomplete transpots have been removed. This makes bad transpot filtering a very useful post-processing for removing noise and improving the results presented to the user.

5.3. Merging variants increases diversity

As mentioned in Section 3.2, a significant number of translations that remain after filtering spurious ones are variants of the same canonical translation. Thus, merging these variants is necessary to avoid displaying many translations that are closely related.

The last line of Table VI reports the results when applying our merging process, after filtering bad transpots with the best classifier found in Section 5.2. Because merging variants applies only to the translation

Table VII. The results (in %) on TEST obtained using the SRF method and the C-HMM-bi algorithm. All the translations are filtered and merged.

	F-measure	MAP_5	MAP_{10}	MAP
before SRF	79.6	45.5	48.7	51.3
after SRF	**80.1**	**45.6**	**48.8**	**51.3**

task, only MAP scores are presented. When comparing the quality of the top translations obtained before (line 2) and after (line 3) merging, it appears that a large gain of almost 10 points is induced by this post-processing for both PBM and C-HMM-bi, and for the three MAP scores. This implies that the reference translations are proposed earlier in the transpot list when grouping similar variants. Therefore, the interest for the user is twofold: the noise is reduced and more relevant translations are proposed at the top of the list, which increases the diversity of suggested translations.

5.4. Relevance feedback slightly improves transpots

To our disappointment, the PRF method described in Section 3.3.1 did not improve the quality of the results for both tasks. A manual inspection shows that PRF discards numerous incorrect translations but introduces more noise than new translations among the top ones.

Using instead the SRF method with our best transpotting method C-HMM-bi[5] slightly improves the F-measure for the transpotting task (Table VII, column 1) and the MAP for the translation task (Table VII, columns 2, 3 and 4). Although SRF mainly acts upon rare transpots, interestingly the substitution of transpots tends to promote better translations among the top responses, like the correct translations qualité de vie and façon d'être in case of Table II.

5.5. Manual human assessment

In order to obtain a qualitative feedback upon the translations suggested by our systems, further experiments were conducted with human assessment.

[5] In order to keep the same transpotting algorithm in the two stages of relevance feedback, we did not apply the SRF to PBM.

5.5.1. *Protocol*

Seven judges were asked to rate the relevance of translations proposed by three transpotting methods: `PBM`, `C-HMM-bi` and `C-HMM-bi` post-processed by `SRF`. For each query, the union of the top 10 results output by these methods was displayed simultaneously and in a random order to each rater. They had to label the quality of each translation with one of the four tags: "clearly good", "fairly good", "quite bad" or "clearly bad". Two sets of 100 queries were built for annotation: $\mathcal{Q}_{\text{freq}}$ is made of frequent queries occurring in at least 1,000 sentence pairs of the TM, while $\mathcal{Q}_{\text{rare}}$ corresponds to rare queries occurring in at most 5 sentence pairs.

Each query of the two data sets was annotated by three judges. Each of the four classes was associated with a relevance score ranging from 0 (for "clearly bad") to 3 (for "clearly good"). Finally, each pair (query, translation) was associated with the average of the three judges' scores.

This annotation turned out to be more complicated than the one we conducted for evaluating erroneous transpots (see Section 3.1). This was confirmed by the Fleiss inter-annotator agreement; kappas of 0.46 and 0.60 were obtained for $\mathcal{Q}_{\text{freq}}$ and $\mathcal{Q}_{\text{rare}}$ respectively, which suggests a moderate agreement beyond chance (Landis and Koch, 1977). However, the differences in annotations usually occur between tags close in the ranking scale.

In order to compare the top k results provided by the three methods for each query q, the *discounted cumulative gain* $DCG(q,k)$ is computed as follows:

$$DCG(q,k) = \sum_{m=1}^{k} \frac{2^{R(q,m)} - 1}{\log_2(1+m)} \tag{13}$$

where $R(q,m)$ is the average relevance score set by the raters for the m^{th} translation of q suggested by one of the three methods to evaluate. This metrics summarizes the relevance score associated with several translations of a given query and reduces the weight of each response with its rank. It is often used in IR for situations of non-binary notions of relevance (Manning et al., 2008).

5.5.2. *Results*

In order to measure the quality perceived by the judges, we averaged the scores provided by the three annotators for the top translations returned by each transpotting method. With a value around 2.7 (for a maximum value of 3.0), this score turns out to be very high, for the three methods and on both $\mathcal{Q}_{\text{freq}}$ and $\mathcal{Q}_{\text{rare}}$. As expected, this score decreases when computed from the translations returned at a lower

Table VIII. Comparison of 3 transpotting systems on $\mathcal{Q}_{\text{freq}}$ and $\mathcal{Q}_{\text{rare}}$ using the DCG metrics (in %). Each cell at line i and column j returns the number of queries, out of 100, in which the method of line i is better than the method of column j.

	>	PBM	C-HMM-bi	C-HMM-bi+SRF
$\mathcal{Q}_{\text{freq}}$	PBM	-	45	46
	C-HMM-bi	51	-	27
	C-HMM-bi+SRF	49	23	-
$\mathcal{Q}_{\text{rare}}$	PBM	-	3	3
	C-HMM-bi	26	-	2
	C-HMM-bi+SRF	26	4	-

rank, which indicates that ordering translations with regard to their frequency in the TM is correct. Nevertheless, the score is still higher than 2.0 for the translations selected at rank 5 for the three systems, and higher than 1.7 when considering the following ranks.

In order to obtain a more fine-grained comparison between systems, DCG was computed from the top 10 translations output by each system. Table VIII reports the number of queries in which one method outperforms another. For example, the cell displayed at line 1 and column 2 means that for 45 queries of $\mathcal{Q}_{\text{freq}}$ PBM obtained a better DCG than C-HMM-bi. The three methods do not have the same behavior for some queries, especially rare ones. Indeed, PBM gave no output for 8 out of 100 queries of $\mathcal{Q}_{\text{freq}}$ at rank 10 and for 37 out of 100 queries of $\mathcal{Q}_{\text{rare}}$ while C-HMM-bi+SRF was able to suggest a translation for all these queries. Consequently, for a fair comparison between PBM and the two other systems, only the queries where PBM gave at least one response were considered.

The results of Table VIII support the equivalence of performance obtained by C-HMM-bi with or without SRF. They outperform each other for around the same number of queries: 27 vs. 23 on $\mathcal{Q}_{\text{freq}}$, and 2 vs. 4 on $\mathcal{Q}_{\text{rare}}$. On the other hand, the comparison of PBM and C-HMM-bi shows more differences. C-HMM-bi outperforms PBM for 51 queries on $\mathcal{Q}_{\text{freq}}$ and for 26 queries on $\mathcal{Q}_{\text{rare}}$, when the contrary only happens for 45 and 3 queries respectively. This significant improvement of C-HMM-bi with respect to PBM is mainly explained by the higher number of translations retrieved per query on average for $\mathcal{Q}_{\text{rare}}$, which tends to improve DCG.

5.6. Methods retained for the new TransSearch

The numerous evaluations we conducted exhibit the domination of two approaches: `PBM` and `C-HMM-bi`. The transpotting algorithm comparison reveals a significant improvement in F-measure when taking into account a contiguity constraint. The enhancement observed in terms of MAP and the preference of human judges towards `C-HMM-bi` leads us to choose this transpotting method for the new TransSearch system.

Besides, our experiments show the interest of filtering bad transpots and merging close translation variants in order to improve the relevance of the top translations. Therefore, we included these two post-processes in TransSearch. As far as the pseudo-relevance feedback is concerned, we decided to ignore it since human evaluation revealed its limited effect to improve the results.

6. Related work

6.1. Word alignment

Since the seminal works of Brown et al. (1993), IBM word alignment models have become the de facto standard in the field of SMT, thanks in part to the open source toolkit Giza++. Several extensions of these models have been proposed in the literature.

Toutanova et al. (2002) introduced modifications to the standard HMM alignment model (Vogel et al., 1996), among which the use of POS tags for smoothing transfer probabilities. Moore (2004) proposed three modifications to the standard EM algorithm used to train IBM model 1, in particular the smoothing of transfer probabilities: it corrects the tendency of this model to align rare source words with too many target words. This issue is taken into account more elegantly by the joint training of asymmetrical word alignment models (one model per translation direction) proposed by Liang et al. (2006). Deng and Byrne (2005) also proposed an extension of the HMM alignment model which compares similarly to IBM model 4, while being more tractable to train.

More recently, Fraser and Marcu (2007) defined a generative model designed for non-consecutive many-to-many word alignment. Words are decomposed into head and non-head words and linked according to syntactic dependencies. Then head words are aligned. The model is trained either in an unsupervised way close to the training of IBM Model 4, or in a semi-supervised way. For the latter training procedure, the authors report interesting results in alignment metrics.

Another line of research investigated the use of a word-aligned bitext in order to improve word alignment. Cherry and Lin (2003) proposed a

model that takes source and target sentences as given and maximizes the probability of links between source and target words, while IBM models maximize the probability that a source sentence generates a target sentence and their alignment. Their model requires the labeling of dependency relations between target words using a syntactic parser (Lin, 1998). Following this work, different supervised discriminative models have been proposed (Ittycheriah and Roukos, 2005; Moore et al., 2006; Blunsom and Cohn, 2006). Various features are computed in order to choose the best sentence alignment. The weights of these features are optimized using the word-aligned bitext. These models report good results according to word alignment metrics such as alignment error rate (AER).

While all these models are very promising, IBM, HMM and phrase-based models remain the mainstream models used by the SMT community. None of the models described above imposed itself as a new standard for word alignment. The use of either a small supervised word-aligned bitext or a syntactic parser is not part of the TransSearch project's road map. Both resources run up against maintainability issues for a commercial application such as TransSearch.

All experiments presented in this paper use the French–English language pair with the Canadian Hansards corpus, but our commercial application is expected to process other language pairs as well as corpora related to other domains (legal, medical, technical, etc.). A model relying on a word-aligned bitext would require us to label manually a new bitext for each new domain corpus. Without this manual annotation, it is unsure how such models would behave in the long-run. Further, a model relying on a parser would require the use of new parsers for each language pair.

In the future, we intend to compare some of the aforementioned generative models to our `C-HMM-bi` approach. Since this approach handles bidirectionality plus a contiguity constraint, and given the results presented in this paper, we are confident of its robustness and we do not expect significant differences compared to the aforementioned models.

6.2. Bilingual concordancers

As mentioned in Section 1, some recent systems try to take advantage of word alignment in order to develop new functionalities, but with only a limited evaluation of their results which does not render a clear judgment on their performance.

Wu et al. (2003) developed a web-based English–Chinese concordancer that highlights the transpots in pairs of sentences in which the query occurs. The tool was mainly designed for second language learn-

ers who can submit queries which are single words, phrases, expressions or even full sentences. The alignment at the word or phrase level relies on the spotting of specific part-of-speech patterns, learned from the idioms and the collocations of an English–Chinese dictionary. Phrases matching these patterns are extracted from the aligned sentences and selected according to a cross-linguistic statistical association criterion.

Callison-Burch et al. (2005) proposed the Linear B system that is currently available for some language pairs between Arabic, Chinese and 7 European languages. The authors use bitexts to compute a phrase table for each language pair. The phrase table is indexed using a suffix array data structure that permits the efficient search of phrase pairs. When the user submits a query, all sentence pairs whose source sentence contains the query are returned. Then the phrase table enables the discovery of the target phrase that best matches in each target sentence according to the given query. In fact, this system is very similar to our `PBM` method. Contrarily to TransSearch, this system does not incorporate a classifier to filter spurious transpots or a method to group similar translation variants. Besides, a limited evaluation was conducted with 120 queries and a TM of 50,000 sentence pairs.

Kockaert et al. (2007) describe a TM system for legal phraseology and terminology in the Belgian national languages (Dutch, French and German). Their experimental tool is based on bilingual resources (dictionaries, lists linking word forms to their lemmas, and stop lists) and cognates to align fragments of bilingual texts. Their automatic alignment method is used to approximate the position of the translation of a query inside sentence pairs of the TM.

7. Conclusion

This paper describes the enhancement of the bilingual concordancer TransSearch using a word alignment functionality. Interestingly, this transforms the nature of the application: it now behaves like a translation finder with a concordancer feature. The application goes beyond a bilingual dictionary thanks to its ability to translate phrases while providing their contexts of occurrence.

We studied and compared several algorithms for the word alignment task we call transpotting. The methods relying either on bidirectional HMMs enhanced with a contiguity constraint or on the standard Moses phrase table obtain the best overall results. While results are quite similar for the transpotting task, the former method provides better results for the translation task, *i.e.* when considering the whole application as a search engine.

Three post-processing methods for improving transpotting results have been studied: a supervised classifier filters out bad translations efficiently; merging close translation variants avoids presenting the user redundant translations and improves the relevance of the results; using pseudo-relevance feedback corrects some transpotting errors, although it only allows for minor improvement.

Most of the experiments were performed using about 2,000 queries and a very large TM of 8.3 million sentence pairs. This amount of data ensures confidence in the results, themselves confirmed by the manual evaluation which takes into account the users' point of view.

The work presented in this paper is implemented in the new version of TransSearch developed in a joint partnership between the authors' laboratory and Terminotix.[6] For the time being, this version gives access to the Canadian Hansards and the Canadian Federal Court Judgments corpora. Only the French–English language pair is available, but the implementation of more language pairs is under consideration. We are confident that users will benefit from this new CAT tool and that it will create new opportunities for its application.

Several of the methods we integrated in our CAT tool could be used in SMT. The `C-HMM-bi` method is used for transpotting in this paper, but it would be suitable for use in building a phrase table by aligning source phrases of reasonable length, instead of focusing on user queries, in order to find interesting target translations for any phrase. This method could also compute a feature based on the contiguity constraint and associated with each phrase pair in a phrase table. The classifiers we used for filtering could also be used for pruning a phrase table, which turns out to improve translation quality (Johnson et al., 2007). Pseudo-relevance feedback considers the occurrences of a phrase as sharing common information rather than being independent. Therefore, it might be relevant to integrate it to the phrase acquisition process of an SMT model similar to that of Marcu and Wong (2002).

Finally, the methods we propose could be used to design a phrase-based TM system which could amount to a full SMT system (Simard, 2003a; Owczarzak et al., 2006; Langlais and Gotti, 2006). After the submission of a text to translate to the system, its source sentences should be properly segmented. For this, a chunker may be used: it would build segments quite similar to user queries studied in this paper and the resulting phrases could be searched in the TM. Because the system would process phrase pairs, rather than sentence pairs in a classical TM, its translation capability would be dramatically enhanced.

[6] `http://www.terminotix.com`

Acknowledgements

The authors would like to thank Fabrizio Gotti and Elliott Macklovitch for their contribution to this work and Jacques Steinlin for programming the user interface of the new prototype. We also thank Gilles Gamas and Micheline Cloutier from Terminotix who supported this research via their contribution to a Collaborative Research and Development (CRD) grant from the Natural Science and Engineering Research Council (NSERC) of Canada.

References

Bishop, C. M.: 1995, *Neural Networks for Pattern Recognition.* New York, NY, USA: Oxford University Press.

Blunsom, P. and T. Cohn: 2006, 'Discriminative Word Alignment with Conditional Random Fields'. In: *21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL).* Sydney, Australia, pp. 65–72.

Breiman, L.: 1996, 'Bagging Predictors'. *Machine Learning* **24**(2), 123–140.

Breiman, L.: 2001, 'Random Forests'. *Machine Learning* **45**(1), 5–32.

Brown, P., V. Della Pietra, S. Della Pietra, and R. Mercer: 1993, 'The Mathematics of Statistical Machine Translation: Parameter Estimation'. *Computational Linguistics* **19**(2), 263–311.

Callison-Burch, C., C. Bannard, and J. Schroeder: 2005, 'A Compact Data Structure for Searchable Translation Memories'. In: *10th European Conference of the Association for Machine Translation (EAMT).* Budapest, Hungary, pp. 59–65.

Casacuberta, F., J. Civeria, E. Cubel, A. L. Lagarda, G. Lapalme, E. Macklovitch, and V. Enrique: 2009, 'Human interaction for high-quality machine translation'. *Communications of the ACM* **52**(10), 135–138.

Cherry, C. and D. Lin: 2003, 'A Probability Model to Improve Word Alignment'. In: *41st Annual Meeting of the Association for Computational Linguistics (ACL).* Sapporo, Japan, pp. 88–95.

Cristianini, N. and J. Shawe-Taylor: 2000, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods.* Cambridge, UK: Cambridge University Press.

Croft, W. and D. Harper: 1979, 'Using Probabilistic Models of Information Retrieval without Relevance Information'. *Journal of Documentation* **35**(4), 285–295.

Deng, Y. and W. Byrne: 2005, 'HMM Word and Phrase Alignment for Statistical Machine Translation'. In: *Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT-EMNLP).* Vancouver, British Columbia, Canada, pp. 169–176.

Fleiss, J. L., B. Levin, and M. C. Pai: 2003, *Statistical Methods for Rates and Proportions.* New York, NY, USA: John Wiley & Sons, 3rd edition.

Foster, G., P. Isabelle, and P. Plamondon: 1997, 'Target-Text Mediated Interactive Machine Translation'. *Machine Translation* **12**, 175–194.

Fraser, A. and D. Marcu: 2007, 'Getting the Structure Right for Word Alignment: LEAF'. In: *Joint Conference on Empirical Methods in Natural Language Process-*

ing and Computational Natural Language Learning (EMNLP-CoNLL). Prague, Czech Republic, pp. 51–60.

Freund, Y. and R. Schapire: 1996, 'Experiments with a New Boosting Algorithm'. In: *13th International Conference on Machine Learning (ICML)*. Bari, Italy, pp. 148–156.

Ittycheriah, A. and S. Roukos: 2005, 'A Maximum Entropy Word Aligner for Arabic-English Machine Translation'. In: *Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT-EMNLP)*. Vancouver, British Columbia, Canada, pp. 89–96.

Johnson, H., J. Martin, G. Foster, and R. Kuhn: 2007, 'Improving Translation Quality by Discarding Most of the Phrasetable'. In: *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Prague, Czech Republic, pp. 967–975.

Kando, N., K. Kuriyama, and M. Yoshioka: 2001, 'Information Retrieval System Evaluation using Multi-Grade Relevance Judgments: Discussion on Averageable Single-Numbered Measures'. *(in Japanese) IPSJ SIG Notes* **FI-63**, 105–112.

Kittler, J., M. Hatef, R. P. Duin, and J. Matas: 1998, 'On Combining Classifiers'. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20**(3), 226–239.

Kockaert, H. J., T. Vanallemeersch, and F. Steurs: 2007, 'Term-Based Context Extraction in Legal Terminology : A Case Study in Belgium'. In: *International Conference on Current Trends in Terminology*. Szombathely, Hungary.

Koehn, P. and B. Haddow: 2009, 'Interactive Assistance to Human Translators using Statistical Machine Translation Methods'. In: *MT Summit XII, The twelfth Machine Translation Summit*. Ottawa, Ontario, Canada.

Koehn, P., H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst: 2007, 'Moses: Open Source Toolkit for Statistical Machine Translation'. In: *45th Annual Meeting of the Association for Computational Linguistics (ACL), Companion Volume*. Prague, Czech Republic, pp. 177–180.

Kuhn, R. and R. De Mori: 1990, 'A Cache-Based Natural Language Model for Speech Recognition'. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12**(6), 570–583.

Landis, J. R. and G. G. Koch: 1977, 'The Measurement of Observer Agreement for Categorical Data'. *Biometrics* **33**, 159–174.

Langlais, P.: 1997, 'A System to Align Complex Bilingual Corpora'. Technical report, CTT, KTH, Stockholm, Sweden.

Langlais, P. and F. Gotti: 2006, 'EBMT by Tree-Phrasing'. *Machine Translation* **20**(1), 1–23. Special Issue on Example-Based Machine Translation.

Liang, P., B. Taskar, and D. Klein: 2006, 'Alignment by Agreement'. In: *Human Language Technology Conference of the North American Association for Computational Linguistics, Main Conference (HLT-NAACL)*. New York, NY, USA, pp. 104–111.

Lin, D.: 1998, 'Dependency-Based Evaluation of MINIPAR'. In: *LREC Workshop on the Evaluation of Parsing Systems*. Granada, Spain, pp. 48–56.

Macklovitch, E.: 2006, 'TransType2: The Last Word'. In: *5th International Conference on Language Resources and Evaluation (LREC)*. Genoa, Italy, pp. 167–172.

Macklovitch, E., G. Lapalme, and F. Gotti: 2008, 'TransSearch: What are translators looking for?'. In: *8th Conference of the Association for Machine Translation in the Americas (AMTA)*. Waikiki, Hawai'i, USA, pp. 412–419.

Macklovitch, E., M. Simard, and P. Langlais: 2000, 'TransSearch: A Free Translation Memory on the World Wide Web'. In: *2nd International Conference on Language Resources and Evaluation (LREC)*. Athens, Greece, pp. 1201–1208.

Manning, C. D., P. Raghavan, and H. Schütze: 2008, *Introduction to Information Retrieval*, Chapt. Evaluation in Information Retrieval, pp. 151–175. New York, NY, USA: Cambridge University Press.

Marcu, D. and W. Wong: 2002, 'A Phrase-Based, Joint Probability Model for Statistical Machine Translation'. In: *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Philadelphia, PA, USA, pp. 133–139.

Moore, R. C.: 2004, 'Improving IBM Word Alignment Model 1'. In: *42nd Meeting of the Association for Computational Linguistics (ACL), Main Volume*. Barcelona, Spain, pp. 518–525.

Moore, R. C., W.-T. Yih, and A. Bode: 2006, 'Improved Discriminative Bilingual Word Alignment'. In: *21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL)*. Sydney, Australia.

Och, F. and H. Ney: 2003, 'A Systematic Comparison of Various Statistical Alignment Models'. *Computational Linguistics* **29**(1), 19–51.

Owczarzak, K., B. Mellebeek, D. Groves, J. Van Genabith, and A. Way: 2006, 'Wrapper Syntax for Example-Based Machine Translation'. In: *7th Conference of the Association for Machine Translation in the Americas (AMTA)*. Boston, MA, USA, pp. 148–155.

Rocchio, J.: 1971, *Relevance Feedback in Information Retrieval*, Chapt. 14, pp. 313–323. Upper Saddle River, NJ, USA: Prentice-Hall Inc.

Sakai, T.: 2004, 'New Performance Metrics Based on Multigrade Relevance: Their Application to Question Answering'. In: *4th National Institute of Informatics Test Collection for IR (NTCIR) Workshop*. Tokyo, Japan.

Simard, M.: 2003a, 'Mémoires de traduction sous-phrastiques'. Ph.D. thesis, Université de Montréal, Québec, Canada.

Simard, M.: 2003b, 'Translation Spotting for Translation Memories'. In: *HLT-NAACL 2003 Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and beyond*. Edmonton, Alberta, Canada, pp. 65–72.

Toutanova, K., H. T. Ilhan, and C. D. Manning: 2002, 'Extensions to HMM-Based Statistical Word Alignment Models'. In: *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Philadelphia, PA, USA, pp. 87–94.

Véronis, J. and P. Langlais: 2000, *Evaluation of Parallel text Alignment Systems — The Arcade Project.*, Chapt. 19, pp. 369–388. Dordrecht, The Netherlands: Kluwer Academic Publisher.

Vogel, S.: 2005, 'PESA: Phrase Pair Extraction as Sentence Splitting'. In: *10th Machine Translation Summit*. Phuket, Thailand, pp. 251–258.

Vogel, S., H. Ney, and C. Tillmann: 1996, 'HMM-Based Word Alignment in Statistical Translation'. In: *16th International Conference on Computational Linguistics (COLING)*. Copenhagen, Denmark, pp. 836–841.

Wu, J.-C., K. C. Yeh, T. C. Chuang, C.-L. Tao-Yuan, W.-C. Shei, and J. S. Chang: 2003, 'TotalRecall: A Bilingual Concordance for Computer Assisted Translation and Language Learning'. In: *41st Annual Meeting on Association for Computational Linguistics (ACL), Companion Volume*. Sapporo, Japan, pp. 201–204.