Improving a general-purpose Statistical Translation Engine by Terminological lexicons

Philippe Langlais
RALI / DIRO / Université de Montréal
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada, H3C 3J7
email:felipe@iro.umontreal.ca

Abstract

The past decade has witnessed exciting work in the field of Statistical Machine Translation (SMT). However, accurate evaluation of its potential in real-life contexts is still a questionable issue.

In this study, we investigate the behavior of an SMT engine faced with a corpus far different from the one it has been trained on. We show that terminological databases are obvious resources that should be used to boost the performance of a statistical engine. We propose and evaluate a way of integrating terminology into a SMT engine which yields a significant reduction in word error rate.

1 Introduction

SMT mainly became known to the linguistic community as a result of the seminal work of Brown et al. (1993b). Since then, many researchers have invested effort into designing better models than the ones proposed in the aforementioned article and several new exciting ways have been suggested to attack the problem.

For instance, Vogel et al. (1996) succeeded in overcoming the independence assumption made by IBM models by introducing order-1 Hidden Markov alignment models. Och et al. (1999) described an elegant way of integrating automatically acquired probabilistic templates into the translation process, and Nießen and Ney (2001) did the same for morphological information.

Radically different statistical models have also been proposed. (Foster, 2000) investigated maximum entropy models as an alternative to the so-called noisy-channel approach. Very recently, Yamada and Knight (2001) described a model in which the noisy-channel takes as input a parsed sentence rather than simple words.

While many of these studies include intensive evaluation sections, it is not always easy to determine exactly how well statistical translation can do on a given task. We know that on a specific task of spoken language translation, Wang (1998) provided evidence that SMT compared favorably to a symbolic translation system; but as mentioned by the author, the comparison was not totally fair.

We do not know of any studies that describe extensive experiments evaluating the adequacy of SMT in a real translation environment. We prefer not to commit ourselves to defining what a real translation task is; instead, we adopt the conservative point of view that a viable translation engine (statistical or not) is one that copes with texts that may be very different in nature from those used to train it.

This fairly general definition suggests that adaptativity is a cornerstone of a successful SMT engine. Curiously enough, we are not aware of much work on adaptative SMT, despite the tremendous amount of work done on adaptative statistical language modeling.

In this paper, we propose to evaluate how a statistical engine behaves when translating a very domain specific text which is far different from the corpus used to trained both our translation and language models. We first describe our translation engine. In section 3, we quantify and analyse the performance deterioration of an SMT engine trained on a broad-based corpus (the Hansard) when used to translate a domain specific text (in this study, a manual for military snipers). In section 4, We then suggest a simple but natural way of improving a broad-based SMT engine; that is, by opening the engine to available terminological resources. In section 5, we report on the improvement we observed by implementing our proposed approach. Finally,

in section 6 we discuss other approaches we feel can lead to more robust translation.

2 Our statistical engine

2.1 The statistical models

In this study, we built an SMT engine designed to translate from French to English, following the noisy-channel paradigm first described by (Brown et al., 1993b). This engine is based on equation 1, where $e_1^{\hat{I}}$ stands for the sequence of $\hat{I}$ English target words to be found, given a French source sentence of J words f_1^J:

$$e_1^{\hat{I}} = \underset{I,e_1^I}{\operatorname{argmax}} \underbrace{P(e_1^I)}_{\text{language}} \cdot \underbrace{P(f_1^J|e_1^I)}_{\text{translation}} \qquad (1)$$

To train our statistical models, we assembled a bitext composed of 1.6 million pairs of sentences that were automatically aligned at the sentence level. In this experiment, every token was converted into lowercase before training.

The language model we used is an interpolated trigram we trained on the English sentences of our bitext. The perplexity of the resulting model is fairly low – 65 –, which actually reflects the fact that this corpus contains many fixed expressions (*e.g* `pursuant to standing order`).

The inverted translation model we used is an IBM2-like model: 10 iterations of IBM1-training were run (reducing the perplexity of the training corpus from 7776 to 90), followed by 10 iterations of IBM2-training (yielding a final perplexity of 54). We further reduced the number of transfer parameters (originally 34 969 331) by applying an algorithm described in Foster (2000); this algorithm basically filters in the pairs of words with the best gain, where gain is defined as the difference in perplexity — measured on a held-out corpus — of a model trained with this pair of words and a model trained without. In this experiment, we worked with a model containing exactly the first gain-ranked million parameters. It is interesting to note that by doing this, we not only save memory, and therefore time, but also obtain improvments in terms of perplexity and overall performance[1].

[1]On a translation task from French to English on Hansard sentences, we observed a reduction in word error rate of more than 3% with the reduced model.

2.2 The search algorithm

The maximum operation in equation 1, also called search or decoding, involves a length model. We assume that the length (counted in words) of French sentences that translate an English sentence of a given length follow a normal distribution.

We extended the decoder described by Nießen et al. (1998) to a trigram language model. The basic idea of this search algorithm is to expand hypotheses along the positions of the target string while progressively covering the source ones. We refer the reader to the original paper for the recursion on which it relies, and instead give in Figure 1 a sketch of how a translation is built. An hypothesis h is fully determined by four parameters: its source (j) and target (i) positions of the last word (e), and its coverage (c). Therefore, the search space can be represented as a 4-dimension table, each item of which contains backtracking information (f for the fertility of e, b_j and b_w for the source position and the target word we should look at to backtrack) and the hypothesis score (*prob*).

We know that better alignment models have been proposed and extensively compared (Och and Ney, 2000). We must however point out that the performance we obtained on the HANSARD corpus (see Section 3) is comparable to the rates published elsewhere on the same kind of corpus. In any case, our goal in this study is to compare the behavior of a SMT engine in both friendly and adverse situations. In our view, the present SMT engine is suitable for such a comparative study.

2.3 Tuning the decoder

The decoder has been tuned in several ways in order to reduce its computations without detrimentally affecting the quality of its output. The first thing we do when the decoder receives a sentence is to compute what we call an *active vocabulary*; that is, a collection of words which are likely to occur in the translation. This is done by ranking for each source word the target words according to their non normalized posterior likelihood (that is $\operatorname{argmax}_e p(f|e)p(e)$, where $p(e)$ is given by a unigram target language model, and $p(f|e)$ is given by the transfer

Input: $f_1 \dots f_j \dots f_J$

Initialize the search space table *Space*
Select a maximum target length: I_{max}
Compute the active vocabulary

// Fill the search table recursively:
for all target position $i = 1, 2, \dots, I_{max}$ **do**
  prune($i-1$);
  for all alive hyp. $h = Space(i, j, c, e)$ **do**
    $uv \leftarrow$ History(h);
    $zones \leftarrow$ FreeSrcPositions(h);
    $bestWords \leftarrow$ NBestTgtWords(uv);
    for all w in $bestWords$ **do**
      $prob \leftarrow$ Score(h) + $\log p(w|uv)$;
      setIfBetter($i, j, c, b, prob, 0, j, v$);
      for all free source position d **do**
        $s \leftarrow prob$;
        for all $f \in [1, f_{max}]$ / $d + f - 1$ is free **do**
          $s+ = \log a(i|d, J) + \log t(f_d|e_i)$;
          setIfBetter($i, d, c+f, w, s, f, j, w$);

// Find and return the best hypothesis if any
$max_s \leftarrow -\infty$
for all $i \in [1, I_{max}]$ **do**
  for all alive hyp. $h = Space(i, j, c, e)$ **do**
    $s \leftarrow Score(h) + \log p(i|J)$;
    if $((c == J)$ and $(s > max_s))$ **then**
      $max_s \leftarrow s$
      $\langle max_i, max_j, max_e \rangle \leftarrow \langle i, j, e \rangle$
if $(max_s! = \infty)$ **then**
  Return $Space(max_i, max_j, J, max_e)$;
else
  Failure

Output: $e_1 \dots e_i \dots e_{max_i}$

Figure 1: Sketch of our decoder. $FreeSrcPositions$ returns the source positions not already associated to words of h; $NBestTgtWords$ returns the list of words that are likely to follow the last bigram uv preceeding e according to the language model; and $setIfBetter(i, j, c, e, p, f, b_j, b_w)$ is an operator that memorizes an hypothesis if its score (p) is greater than the hypothesis already stored in $Space(i, j, c, e)$. a and t stands for the alignment and transfert distributions used by IBM2 models.

probabilities of our inverted translation model) and keeping for each source word at most a target words.

Increasing a raises the coverage of the active vocabulary, but also slows down the translation process and increases the risk of admitting a word that has nothing to do with the translation. We have conducted experiments with various a-values, and found that an a-value of 10 offers a good compromise.

As mentioned in the block diagram, we also prune the space to make the search tractable. This is done with relative filtering as well as absolute thresholding. The details of all the filtering strategies we implemented are however not relevant to the present study.

3 Performances of our SMT engine

3.1 Test corpora

In this section we provide a comparison of the translation performances we measured on two corpora. The first one (namely, the HANSARD) is a collection of sentences extracted from a part of the Hansard corpus we did not use for training. In particular, we did not use any specific strategy to select these sentences so that they would be closely related to the ones that were used for training.

Our second corpus (here called SNIPER) is an excerpt of an army manual on sniper training and deployment that was used in an EARLIER study (Macklovitch, 1995). This corpus is highly specific to the military domain and would certainly prove difficult for any translation engine not specifically tuned to such material.

3.2 Overall performance

In this section, we evaluate the performance of our engine in terms of sentence- and word- error rates according to an oracle translation[2]. The first rate is the percentage of sentences for which the decoder found the exact translation (that is, the one of our oracle), and the word error rate is computed by a Levenstein distance (counting the same penalty for both *insertion*, *deletion* and *substitution* edition operations). We realize that these measures alone are not sufficient for a serious evaluation, but we were re-

[2]Both corpora have been published in both French and English, and we took the English part as the gold standard.

luctant in this experiment to resort to manual judgments, following for instance the protocol described in (Wang, 1998). Actually a quick look at the degradation in performance we observed on SNIPER is so clear that we feel these two rates are informative enough !

Table 1 summarizes the performance rates we measured. The WER is close to 60% on the HANSARD corpus and close to 74% on SNIPER; source sentences in the latter corpus being slightly longer on average (21 words). Not a single sentence was found to be identical to the gold standard translation on the SNIPER corpus [3].

corpus	*nbs*	$\|length\|$	SER	WER
HANSARD	1038	$\langle 16.2, 7.8 \rangle$	95.6	59.6
SNIPER	203	$\langle 20.8, 6.8 \rangle$	100	74.6

Table 1: Main characteristics of our test corpora and global performance of our statistical translator without any adjustments. $|length|$ reports the average length (counted in words) of the source sentences and the standard deviation; *nbs* is the number of sentences in the corpus.

3.3 Analyzing the performance drop

As expected, the poor performance observed on the SNIPER text is mainly due to two reasons: the presence of out of vocabulary (OOV) words and the incorrect translations of terminological units.

In the SNIPER corpus, 3.5% of the source tokens and 6.5% of the target ones are unknown to the statistical models. 44% of the source sentences and 77% of the target sentences contain at least one unknown word. In the HANSARD text, the OOV rates are much lower: around 0.5% of the source and target tokens are unknown and close to 5% of the source and target sentences contain at least one OOV words.

These OOV rates have a clear impact on the coverage of our active vocabulary. On the SNIPER text, 72% of the oracle tokens are in the active vocabulary (only 0.5% of the target sentences are fully covered); whilst on HANSARD, 86% of the oracle's tokens are covered (24% of the target sentences are fully covered).

Another source of disturbance is the presence of terminological units (TU) within the text to translate. Table 2 provides some examples of mistranslated TU from the SNIPER text. We also observed that many words within terminological units are not even known by the statistical models. Therefore accounting for terminology is one of the ways that should be considered to reduce the impact of OOV words.

< source term / oracle / translation>
<âme / bore / heart>
<huile polyvalente / general purpose oil / oil polyvalente>
<chambre / chamber / house of common>
<tireur d' élite / sniper / issuer of elite>
<la longueur de la crosse / butt length / the length of the crosse>

Table 2: Examples of mistranslated terminological entries of the SNIPER corpus.

4 Integrating non-probabilistic terminological resources

Using terminological resources to improve the quality of an automatic translation engine is not at all a new idea. However, we know of very few studies that actually investigated this avenue in the field of statistical machine translation. Among them, (Brown et al., 1993a) have proposed a way to exploit bilingual dictionnaries at training time. There may also be cases where domain-specific corpora are available which allow for the training of specialized models that can be combined with the general ones.

Another approach that would not require such material at training time consists in designing an adaptative translation engine. For instance, a cache-based language model could be used instead of our static trigram model. However, the design of a truly adaptative translation model remains a more speculative enterprise. At the very least, it would require a fairly precise location of errors in previously translated sentences; and we know from the ARCADE campaign on bilingual alignments, that accurate word alignments are difficult to obtain (Véronis and Langlais, 2000). This may be even more difficult in situations where errors will in-

[3] The full output of our translation sessions is available at `www-iro.umontreal.ca/~felipe/ResearchOutput/Computerm2002`

volve OOV words.

We investigated a third option, which involves taking advantage – at run time – of existing terminological resources, such as *Termium*[4]. As mentioned by Langlais et al. (2001), one of a translator's first tasks is often terminological research; and many translation companies employ specialized terminologists. Actually, aside from the infrequent cases where, in a given thematic context, a word is likely to have a clearly preferred translation (*e.g.* `bill`/*facture vs* `bill`/*projet de loi*), lexicons are often the only means for a user to influence the translation engine.

Merging such lexicons at run time offers a complementary solution to those mentioned above and it should be a fruitful strategy in situations where terminological resources are not available at training time (which may often be the case). Unfortunately, integrating terminological (or user) lexicons into a probabilistic engine is not a straightforward operation, since we cannot expect them to come with attached probabilities. Several strategies do come to mind, however. For instance, we could credit a translation of a sentence that contains a source lexicon entry in cases it contains an authorized translation. But this strategy may prouve difficult to tune since decoding usually involves many filtering strategies.

The approach we adopted consists in viewing a terminological lexicon as a set of constraints that are employed to reduce the search space. For instance, knowing that `sniper` is a sanctioned translation of *tireur d'élite*, we may require that current hypotheses in the search space associate the target word `sniper` with the three source French words.

In our implementation, we had to slightly modify the block diagram of Figure 1 in order to: 1) forbid a given word e_i from being associated with a word belonging to a source terminological unit, if it is not sanctioned by the lexicon; and 2) add at any target position an hypothesis linking a target lexicon entry to its source counterpart. Whether these hypotheses will survive intact will depend on constraints imposed by the maximum operation (of equation 1) over the full translation.

The score associated with a target entry $e_i^{i'}$ when linked to its source counterpart $f_j^{j'}$ in the latter case is given by:

$$\sum_{k\in[i,i']} \log p(e_k|e_{k-2}e_{k-1}) + \max_{l\in[j,j']} \log(a(k|l,J))$$

The rationale behind this equation is that both the language (p) and the alignment (a) models have some information that can help to decide the appropriateness of an extension: the former knows how likely it is that a word (known or not) will follow the current history[5]; and the latter knows to some extent where the target unit should be (regardless of its identity). In the absence of a better mechanism (e.g. a cache-model should be worth a try) We hope that this will be sufficient to determine the final position of the target unit in a given hypothesis.

5 Results

We considered three terminological lexicons whose characteristics are summarized in Table 3; they essentially differ in terms of number of entries and therefore coverage of the text to translate.

lexicon	nb	coverage	SER	WER
`sniper-1`	33	20/247	99	67.4
`sniper-2`	59	47/299	98	66.2
`sniper-3`	146	132/456	98	64.3

Table 3: Translation performance with different terminological lexicons. *nb* is the number of entries in the lexicon and *coverage* reports the number of different source entries from the lexicon belonging to the text to translate and the total number of their occurrences.

The first lexicon (namely `sniper-1`) contains the 33 entries used in the study of terminological consistency checking described in (Macklovitch, 1995). The second and third lexicons (namely `sniper-2` and `sniper-3`) contain those entries plus other ones added manually after an incremental inspection of the SNIPER corpus.

As can be observed from Table 3, introducing terminological lexicons into the translation engine does improve performance, measured in terms of WER, and this even with lexicons that

[4]See `http://www.termium.com/site/`.

[5]Our trigram model has been trained to provide parameters such as $p(UNK|ab)$.

Source	le **tireur d' élite** voit simultanément les **fils croisés** et l' image (l' objectif) .
Target	the **sniper** sees the **crosshairs** and the image - target - at the same time .
without	*the gunman being same son sit and picture of the hon. members : agreed .*
with	*the **sniper** simultaneously see the **crosshairs** and the image (objective .)*
Source	contrôle de la **détente** .
Target	exercising **trigger** control .
without	*the control of détente .*
with	*control of the **trigger** .*

Table 4: Two examples of translation *with* and *without* a terminological lexicon; TU appear in bold.

cover only a small portion of the text to translate. With the narrow coverage lexicon, we observe an absolute reduction of 7%, and a reduction of 10% with the broader lexicon `sniper-3`. This suggests that adding more entries into the lexicon is likely to decrease WER. In another study (Carl and Langlais, 2002), we investigated whether an automatic procedure designed to detect term variants could improve these performances futher.

Table 4 provides two examples of translation outputs, with and without the help of terminological units. The first one clearly shows that EVEN A few TU (two in this case) may substantially improve the quality of the translation output; (the translation produced without the lexicon was particularly poor in this very case.

Even though terminological lexicons do improve the overall WER figure, a systematic inspection of the outputs produced with TU reveals that the translations are still less faithful to the source text than the translations produced for the HANSARD text. OOV words remain a serious problem.

6 Discussion

In this study, we have shown that translating texts in specific domains with a general[-]purpose statistical engine is difficult. This suggests the need to implementing an adaptative strategy. Among the possible scenarios, we have shown that opening the engine to terminological resources is a natural and efficient way of softening the decoder.

In a similar vein, Marcu (2001) investigated how to combine Example Based Machine Translation (EBMT) and SMT approaches. The author automatically derived from the Hansard corpus what he calls a translation memory: actually a collection of pairs of source and target word sequences that are in a translation relation according to the viterbi alignment run with an IBM4 model that was also trained on the Hansard corpus. This collection of phrases was then merged with a greedy statistical decoder to improve the overall performance of the system.

What this study suggests is that translation memories collected from a given corpus can improve the performance of a statistical engine trained on the same corpus, which in itself is an interesting result. A very similar study but with weaker results is derscribed in (Langlais et al., 2000), in the framework of the TRANSTYPE project. Besides the different metrics the authors used, the discrepancy in performance in these two studies may be explained by the nature of the test corpora used. The test corpus in the latter study was more representative of a real translation task, while the test corpus that Marcu used was a set of around 500 French sentences of no more than 10 words.

Our present study is close in spirit to these last two, except that we do not attack the problem of automatically acquiring bilingual lexicons; instead, we consider it a part of the translator's task to provide such lexicons. Actually, we feel this may be one of the only ways a user has of retaining some control over the engine's output, a fact that professional translators seem to appreciate (Langlais et al., 2001).

As a final remark, we want to stress that we see the present study as a first step toward the eventual unification of EBMT and SMT, and in this respect we agree with (Marcu, 2001). Potentially, of course, EBMT can offer much more than just a simple list of equivalences, like those we used in this study. However, the basic ap-

proach we describe here still holds, as long as we can extend the notion of *constraint* used in this study to include non-consecutive sequences of words. This is a problem we we plan to investigate in future research.

Acknowledgments

I am indebted to Elliott Macklovitch and George Foster for the fruitful orientation they gave to this work.

References

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Meredith J. Goldsmith, Jan Hajic, Robert L. Mercer, and Surya Mohanty. 1993a. But dictionaries are data too. In *Human Language Technology (HLT)*, pages 202–205, Princeton, NJ, march.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993b. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Michael Carl and Philippe Langlais. 2002. Toward an intelligent terminology database as a front-and backend for statistical machine translation. In *COMPUTERM 2002*, Taipei.

George Foster. 2000. A Maximum Entropy / Minimum Divergence translation model. In *Proceedings of the 38th Annual Meeting of the ACL*, pages 37–44, Hong Kong, October.

Philippe Langlais, George Foster, and Guy Lapalme. 2000. Unit completion for a computer-aided translation typing system. In *Proceedings of the 5th Conference on Applied Natural Language Processing (ANLP)*, pages 135–141, Seattle, Washington, May.

Philippe Langlais, George Foster, and Guy Lapalme. 2001. Integrating bilingual lexicons in a probabilistic translation assistant. In *Proceedings of the 8th Machine Translation Summit*, pages 197–202, Santiago de Compostela, Galicia, Spain, September. IAMT.

Elliott Macklovitch. 1995. Can terminological consistency be validated automatically ? Technical report, CITI/RALI, Montréal, Canada.

Daniel Marcu. 2001. Towards a unified approach to memory- and statistical-based machine translation. In *Proceedings of the 39th Annual Meeting of the ACL*, pages 378–385, Toulouse, France.

Sonja Nießen and Hermann Ney. 2001. Toward hierarchical models for statistical machine translation of inflected languages. In *Proceedings of the Workshop on Data Driven Machine Translation yielded at the 39th Annual Meeting of the ACL*, pages 47–54, Toulouse, France.

Sonja Nießen, Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1998. A dp based search algorithm for statistical machine translation. In *Proceedings of the 36th Annual Meeting of the ACL and the 17th COLING*, pages 960–966, Montréal, Canada, August.

Franz Joseph Och and Hermann Ney. 2000. A comparison of alignement models for statistical machine translation. In *Proceedings of the International Conference on Computational Linguistics (COLING) 2000*, pages 1086–1090, Saarbrucken, Luxembourg, Nancy, August.

Franz Josef Och, Christoph Tillman, and Herman Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the 4nd Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 20–28, College Park, Maryland.

Jean Véronis and Philippe Langlais, 2000. *Evaluation of parallel text alignment systems: The ARCADE project*, volume 13, chapter 19, pages 369–388. Parallel Text Processing, Kluwer.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. Hmm-based word alignement in statistical translation. In *Proceedings of the International Conference on Computational Linguistics (COLING) 1996*, pages 836–841, Copenhagen, Denmark, August.

Ye-Yi Wang. 1998. *Grammar Inference and Statistical Machine Translation*. Ph.D. thesis, CMU-LTI, Carnegie Mellon University.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the ACL*, pages 531–538, Toulouse, France.