A Computer-Assisted Translation Tool based on Finite-State Technology[*]

Jorge Civera[♣], Juan M. Vilar[◇], Antonio L. Lagarda[♣], Elsa Cubel[♠], Sergio Barrachina[♡], Francisco Casacuberta,[♣], Enrique Vidal[♠]

[♣]Departamento de Sistemas Informáticos y Computación. Univ. Politècnica de València (Spain)
[♠]Instituto Tecnológico de Informática. Univ. Politècnica de València (Spain)
[◇]Departamento de Lenguajes y Sistemas Informáticos. Univ. Jaume I (Spain)
[♡]Departamento de Ingeniera y Ciencia de Computadores. Univ. Jaume I (Spain)

tt2iti@iti.upv.es

Abstract

The Computer-Assisted Translation (CAT) paradigm tries to integrate human expertise into the automatic translation process. In this paradigm, a human translator interacts with a translation system that dynamically offers a list of translations that best completes the part of the sentence that is being translated. This human-machine sinergy aims at a double goal, to increase translator productivity and ease translators' work. In this paper, we present a CAT system based on stochastic finite-state transducer technology. This system has been developed and assessed on two real parallel corpora in the framework of the European project TransType2 (TT2).

1 Introduction

Information technology advances in modern society have led to the need of more efficient methods of translation. It is important to emphasise that current Machine Translation (MT) systems are not able to produce ready-to-use text. Indeed, MT systems are usually limited to specific semantic domains and the translations provided require post-editing in order to achieve a correct translation.

A way of taking advantage of MT systems is to combine them with the knowledge of a human translator by means of a Computer-Assisted Translation (CAT) system. In a CAT system, the user can amend the translation offered by the MT system, as the system takes into account these corrections to improve its translation. In this type of systems, the translator can work in a more comfortable way because of the greater freedom to make changes at any time while the translation is in progress.

The CAT approach has two important aspects: the models need to provide adequate completions and they have to do so efficiently under usability constrains. To fulfill these two requirements, Stochastic Finite-State Transducers (SFST) have been selected since they have proved to be able to provide adequate translations (Knight & Al-Onaizan, 1998; Amengual et al., 2000; Bangalore & Riccardi, 1995). In addition, efficient parsing algorithms can be easily adapted in order to provide completions.

The rest of the paper is structured as follows. Next section introduces the general setting for MT and finite-state models. In Section 3, the search procedure for interactive translation is explained. Experimental results are presented in Section 4. Finally, some conclusions and future work are exposed in Section 5.

[*]This work has been supported by the European Union under the IST Programme (IST-2001-32091) and the Spanish project TIC2003-08681-C02.

2 Machine translation with finite-state transducers

In a probabilistic framework, given a source sentence $\mathbf{s}$, the goal of MT is to find a target sentence $\hat{\mathbf{t}}$ that:

$$\hat{\mathbf{t}} = \underset{\mathbf{t}}{\operatorname{argmax}} \Pr(\mathbf{t} \mid \mathbf{s}) = \underset{\mathbf{t}}{\operatorname{argmax}} \Pr(\mathbf{t}, \mathbf{s}). \quad (1)$$

The joint distribution $\Pr(\mathbf{t}, \mathbf{s})$ can be modelled by a SFST $\mathcal{T}$ (Picó & Casacuberta, 2001):

$$\hat{\mathbf{t}} = \underset{\mathbf{t}}{\operatorname{argmax}} \Pr(\mathbf{t}, \mathbf{s}) \approx \underset{\mathbf{t}}{\operatorname{argmax}} \Pr_{\mathcal{T}}(\mathbf{t}, \mathbf{s}). \quad (2)$$

It should be noted that the maximisation problem stated above is NP-hard (Casacuberta & Higuera, 2000). Nevertheless, adequate approximations can be obtained by neans of efficient search algorithms, like Viterbi (Viterbi, 1967) for the best path and the Recursive Enumeration Algorithm (REA) (Jiménez & Marzal, 1999) for the n-best paths. SFSTs have been successfully applied to many translation tasks (Amengual et al., 2000; Casacuberta et al., 2004).

A possible way of learning SFSTs from training data is the Grammatical Inference and Alignments for Transducer Inference (GIATI) technique (Casacuberta & Vidal, 2004). Given a finite sample of source-target sentence pairs, it works in three steps:

1. Building training strings: Each training pair is transformed into a single string from an extended alphabet to obtain a new sample of strings. The "extended alphabet" contains words or substrings from source and target sentences coming from training pairs.

2. Inferring a (stochastic) regular grammar: Typically, a smoothed n-gram is inferred from the sample of strings obtained in the previous step.

3. Transforming the inferred regular grammar into a transducer: The symbols associated to the grammar rules are adequately transformed back into source/target symbols.

The transformation of a parallel corpus into a corpus of single sentences is performed with the help of statistical alignments: each word is joined with its translation in the output sentence, creating an "extended word". This joining is done taking care not to invert the order of the output words. The third step is trivial with this arrangement. In our experiments, the alignments are obtained using the GIZA++ software (Och & Ney, 2000), which implements IBM statistical models (Brown et al., 1993).

3 Interactive search

The CAT paradigm introduces a new factor $\mathbf{t}_p$ into the general MT equation (Eq. 1). $\mathbf{t}_p$ represents a prefix of the target sentence obtained as a result of the interaction between the human translator and the MT system.

An example of this interaction is shown in Fig. 1. In each iteration, a prefix ($\mathbf{t}_p$) of the target sentence has somehow been fixed by the human translator in the previous iteration and the CAT system computes its best (or n-best) translation suffix hypothesis ($\hat{\mathbf{t}}_s$) to complete this prefix.

Given $\mathbf{t}_p\hat{\mathbf{t}}_s$, the CAT cycle proceeds by letting the user establish a new, longer acceptable prefix. To this end, he or she has to accept a part ($\mathbf{a}$) of $\mathbf{t}_p\hat{\mathbf{t}}_s$ (or, more typically, just a prefix of $\hat{\mathbf{t}}_s$). After this point, the user may type some keystrokes ($\mathbf{k}$) in order to amend some remaining incorrect parts. Therefore, the new prefix typically encompasses $\mathbf{t}_p$ followed by the accepted part of the system suggestion, $\mathbf{a}$, plus the text, $\mathbf{k}$, entered by the user. Now this prefix, $\mathbf{t}_p\,\mathbf{a}\,\mathbf{k}$, becomes a new $\mathbf{t}_p$, thereby starting a new CAT prediction cycle.

Ergonomics and user preferences dictate exactly when the system can start its new cycle, but typically, it is started after each user-entered word or even after each new user keystroke.

Perhaps the simplest formalization of the process of hypothesis suggestion of a CAT system is as follows. Given a source text $\mathbf{s}$ and a user validated *prefix* of the target sentence $\mathbf{t}_p$, search for a *suffix* of the target sentence that maximises the *a posteriori* probability over all possible suffixes:

ITER-0	$(\mathbf{t}_p)$	( )
ITER-1	$(\hat{\mathbf{t}}_s)$	(*Haga clic para cerrar el diálogo de impresión*)
	$(\mathbf{a})$	(Haga clic)
	$(\mathbf{k})$	(en)
	$(\mathbf{t}_p)$	(Haga clic en)
ITER-2	$(\hat{\mathbf{t}}_s)$	(*ACEPTAR para cerrar el diálogo de impresión*)
	$(\mathbf{a})$	(*ACEPTAR para cerrar el*)
	$(\mathbf{k})$	(cuadro)
	$(\mathbf{t}_p)$	(Haga clic en ACEPTAR para cerrar el cuadro)
FINAL	$(\hat{\mathbf{t}}_s)$	(*de diálogo de impresión*)
	$(\mathbf{a})$	(*de diálogo de impresión*)
	$(\mathbf{k})$	(#)
	$(\mathbf{t}_p \equiv \mathbf{t})$	(Haga clic en ACEPTAR para cerrar el cuadro de diálogo de impresión)

Figure 1: Example of a CAT system interaction to translate into Spanish the English sentence *"Click OK to close the print dialog"* extracted from a printer manual. Each step starts with a previously fixed target language prefix $\mathbf{t}_p$, from which the system suggests a suffix $\hat{\mathbf{t}}_s$. Then the user accepts part of this suffix ($\mathbf{a}$) and types some keystrokes ($\mathbf{k}$), in order to amend the remaining part of $\mathbf{t}_s$. This produces a new prefix, composed by the prefix from the previous iteration and the accepted and typed text, $(\mathbf{a})\,(\mathbf{k})$, to be used as $\mathbf{t}_p$ in the next step. The process ends when the user enters the special keystroke "#". In the final translation, $\mathbf{t}$, all the text that has been typed by the user is underlined.

$$\hat{\mathbf{t}}_s = \underset{\mathrm{t}_s}{\operatorname{argmax}} \Pr(\mathbf{t}_s \mid \mathbf{s}, \mathbf{t}_p)\ . \qquad (3)$$

Taking into account that $\Pr(\mathbf{t}_p \mid \mathbf{s})$ does not depend on $\mathbf{t}_s$, we can write:

$$\hat{\mathbf{t}}_s = \underset{\mathrm{t}_s}{\operatorname{argmax}} \Pr(\mathbf{t}_p\mathbf{t}_s \mid \mathbf{s})\ , \qquad (4)$$

where $\mathbf{t}_p\mathbf{t}_s$ is the concatenation of the given prefix $\mathbf{t}_p$ and a suffix $\mathbf{t}_s$. Eq. 4 is similar to Eq. 1, but here the maximisation is carried out over a set of suffixes, rather than full sentences as in Eq. 1. This joint distribution can be adequately modeled by means of SFSTs (Civera et al., 2004b).

The solution to this maximisation problem has been devised in two phases. The first one copes with the extraction of a word graph $\mathcal{W}$ from a SFST $\mathcal{T}$ given a source sentence $\mathbf{s}$. In a second phase, the search of the best translation suffix (or suffixes) according to the Viterbi approach (Viterbi, 1967) is performed over the word graph $\mathcal{W}$ given a prefix $\mathbf{t}_p$ of the target sentence.

3.1 Word graph derivation

A word graph is a compact representation of all the possible translations that a SFST $\mathcal{T}$ can produce from a given source sentence $\mathbf{s}$ (Civera et al., 2004b, 2004a). In fact, the word graph could be seen as a kind of weighted finite-state automaton in which the probabilities are not normalized.

There are a couple of minor issues to deal with in this construction. On the one hand, the output symbol for a given transition could contain more than one word. In this case, auxiliary states were created to assign only one word for each transition and simplify the posterior search procedure. On the other hand, it is possible to have words in the input sentence that do not belong to the input vocabulary in the SFST. This problem is solved with the introduction of a special generic "unknown word" in the input vocabulary of the SFST.

Intuitively, the word graph generated retains those transitions in the SFST that were compatible with the source sentence along with their transition probability and output symbol(s). Those states that are reached at the end of the parsing process of the source sentence, over the SFST, are considered final states (as well as those states reachable with λ-transitions from them).

Once the word graph is constructed, it can be used to find the best completions for the part of the translation typed by the human translator. Note that the word graph depends only on the input sentence, so it is used repeatedly for finding the completions of all the different prefixes provided by the user.

3.2 Search for n-best translations given a prefix of the target sentence

Ideally, the search problem consists in finding the target suffix $\mathbf{t}_s$ that maximises the *a posteriori* probability given a prefix $\mathbf{t}_p$ of the target sentence and the input sentence $\mathbf{s}$, as described in Eq. 4. To simplify this search, it will be divided into two steps or phases. The first one would deal with the parsing of $\mathbf{t}_p$ over the word graph $\mathcal{W}$. This parsing procedure would end reaching a set of states Q'_p that define paths from the initial state whose associated translations include $\mathbf{t}_p$. To clarify this point, it is important to note that each state q in the word graph defines a set of translation prefixes P_q. This set of translation prefixes is obtained from the concatenation of the output symbols of the different paths that reach this state q from the initial state. Therefore, the set P_q of each state in Q'_p includes $\mathbf{t}_p$. The second phase would be the search of the most probable translation suffix from any of the states in Q'_p. Finally, the complete search procedure extracts a translation from the word graph whose prefix is $\mathbf{t}_p$ and its remaining suffix is the resulting translation suffix $\mathbf{t}_s$.

3.2.1 Error-correcting parsing.

In practice, however, it may happen that $\mathbf{t}_p$ is not exactly present in the word graph $\mathcal{W}$. The solution is not to use $\mathbf{t}_p$ but a prefix $\mathbf{t}'_p$ that is the *most similar* to $\mathbf{t}_p$ in some string distance metric. The metric that will be employed is the well-known minimum edit distance based on three basic edit operations: insertion, substitution and deletion. Therefore, the first phase introduced in the previous paragraph needs to be redefined in terms of the search for those states in $\mathcal{W}$ whose set P_q contains $\mathbf{t}'_p$; that is, the set of states Q'_p. It should be remarked that $\mathbf{t}'_p$ is not unique, but there exist a set of prefixes in $\mathcal{W}$ whose minimum edit distance to $\mathbf{t}_p$ is the same and the lowest possible.

Given a translation prefix $\mathbf{t}_p$, the computation of Q'_p is efficiently carried out by applying an adapted version of the error-correcting algorithm for regular grammars over the word graph $\mathcal{W}$. This algorithm returns the minimum edit cost $c(q)$ with respect to $\mathbf{t}_p$ for each state q in $\mathcal{W}$. To be more precise, this minimum edit cost is the lowest minimum edit cost between $\mathbf{t}_p$ and the set of prefixes P_q of each state q.

Once the set Q'_p has been computed, the search of the most probable translation suffix could be calculated from any of the states in Q'_p.

Furthermore, it may be the case that a user prefix ends in an incomplete word during the interactive translation process. Therefore, it is necessary to start the translation suffix by a word that best completes this unfinished word.

3.2.2 N-best search.

The actual implementation of this CAT system is able to provide a set of different translation suffixes, instead of a single suggestion. To this purpose, an algorithm that searches for the n-best translation suffixes in a word graph is required. Among the n-best algorithms available, the Recursive Enumeration Algorithm (REA) described in (Jiménez & Marzal, 1999) was selected. The main two reasons that support this decision are its simplicity to calculate best paths on demand and its smooth integration with the error-correcting parsing algorithm. Basically, the interaction between these two algorithms, error-correcting and n-best, consists in the supplement of the state q_p by the former, so that the n-best translation suffixes can be calculated from this state by the latter.

The version of REA included in the CAT system, which is being described, stores for each state q in $\mathcal{W}$, the sorted list of current best paths (in the form of next state in the best path) from q to any final state. The length of this sorted list depends on the number of transitions leaving q. During the initialisation of REA, the initial sorted list of best paths for each state is calculated starting from the final states and visiting the rest of states in backward topological order. This last condition imposes a total order in Q' that favours the efficient calculation of the sorted list of best paths. This is so because each state is visited only once, and once the best paths of the preceding states have al-ready been computed.

Then, among the set of states in Q'_p from which the n-best translation suffixes need to be calculated, REA first extracts the 1-best path from the set of states Q'_p, since it was precomputed during REA initialisation. If $n > 1$, then the next best path will be obtained. The next best path can be found among the candidate paths still left in the sorted list of states in Q'_p and the second best path in the 1-best state through the transition traversed in the 1-best path just extracted.
This fact requires the recursive calculation of the second best path (whenever exists) through the states visited in the 1-best path. This same rationale is applied to the calculation of subsequent best paths until n-best different translation suffixes have been obtained or no more best paths can be found.

4 Experimental framework and results

The SFST models introduced in the previous sections were assessed through some series of experiments with two different corpora that were acquired and preprocessed in the framework of the TransType2 (TT2) project (Atos Origin, Instituto Tecnológico de Informática, RWTH Aachen, RALI Laboratory, Celer Soluciones and Société Gamma and Xerox Research Centre Europe, 2001). In this section, these corpora, the assessment metrics and the results are presented.

4.1 XRCE and EU corpora

Two bilingual corpora from different semantic domains were used in the evaluation of the CAT system described. The language pairs involved in the assessment were English/Spanish, English/French and English/German.

The first corpus, namely *XRCE* corpus, was obtained from a miscellaneous set of printer user manuals. Some statistics of this corpus are shown in Table 1.

The English manuals are different in each pair of languages. The size of the vocabulary in the training set is about 25.000 words in most of the language pairs. In the test set, even though all test sets have similar size, perplexity varies abruptly over the different language pairs.

The second dataset was compiled from the Bulletin of the European Union, which exists in the 11 official languages of the European Union and is publicly available on the Internet. This dataset is known as the *EU* corpus. A summary of its features is presented in Table 1.

The size of the vocabulary of this corpus is at least three times larger than that of the *XRCE* corpus. These figures together with the amount of running words and sentences reflect the challenging nature of this task. However, the perplexity of the *EU* test set is similar to that of the *XRCE*. This phenomenon can be intuitively explained through the more uniform grammatical structure of the sentences in the *EU* corpus.

4.1.1 Corpora preprocessing

In order to reduce the corpora complexity, a preprocess module was implemented. In this way, models could be learnt more accurately. Vocabulary size was cut down considerably (up to a 70%), implying an increment in the number of running words (less than 20%). As a result, perplexity decreased up to seven points, which allowed a better transducer inference.

The developed preprocess had three main parts: tokenization, lower case conversion and categorization. Tokenization basically consisted in the separation of the punctuation marks from the words. After that, all the characters were lowercased. Finally, a categorisation of some types of words was carried out. The idea was to replace those words that remain invariable in all the languages with a category label.

This preprocess was also applied in the translation process, since the models were learnt on the preprocessed version of the corpora. Consequently, prefixes written by the user had also to be preprocessed. In addition, a postprocess module was needed to make the translations given by the system legible by the user, undoing all the changes

Table 1: The "XRCE" and "EU" corpora English(En) to/from Spanish(Sp), German(Ge) and French(Fr). Trigrams models were used to compute the test perplexity. (K denotes $\times$1.000, and M denotes $\times$1.000.000).

		XRCE			EU		
		En/Sp	En/Ge	En/Fr	En/Sp	En/Ge	En/Fr
Train	Sent. pairs (K)	56	49	53	214	223	215
	Run. words (M)	0.6/0.7	0.6/0.5	0.6/0.7	5.9/6.6	6.5/6.1	6.0/6.6
	Vocabulary (K)	26/30	25/27	25/37	84/97	87/153	85/91
Test	Sentences (K)	1.1	1.0	1.0	0.8	0.8	0.8
	Run. words (K)	8/9	9/10	11/10	20/23	20/19	20/23
	Perplexity	107/60	93/169	193/135	96/72	95/153	97/71

introduced by the preprocess (i.e. uncategorisating, suitably uppercasing and joining punctuation marks to words).

4.2 Translation quality evaluation

The evaluation metrics used in this work were aimed at estimating the effort reduction, in terms of key strokes and/or mouse actions, that a translator would enjoy when using the CAT system. Specifically, the following three CAT evaluation metrics were considered:

- *Key-Stroke Ratio* (KSR). Number of "key strokes" that should be needed to obtain the reference translation divided by the number of running characters.
- *Mouse-Action Ratio* (MAR). Number of "mouse movements" plus an extra "mouse action" accounting for the acceptance of the final correct translation. A mouse movement is assumed to happen between key strokes which are in non-consecutive positions. It models the effort to position the cursor each time the user would need to amend a part of the system translation.
- *Key-Stroke and Mouse-Action Ratio* (KMSR). KSR plus MAR.

KSR reflects the ratio between the number of key-stroke interactions of a fictitious user when translating a given text using a CAT system compared to the number of keystroke interactions, which this user would need, to translate the same text without using a CAT system. The second measure under consideration is KMSR (the calculation of MAR is straightforward given KSR and KMSR) offers a better approximation to the total amount of work that a translator would be saving when translating using a CAT system.

4.3 Experimental results

These experimental results were obtained with GIATI transducers based on smoothed trigram language models for the *XRCE* corpus and smoothed 5-gram language models for the *EU* corpus (see Table 2).

The translation metrics presented in the previous section were calculated on an independent test set when translating from English into a non-English language, as shown on the left-most column of Table 2. Similar results were obtained when translating from a non-English language into English. Moreover, the results were obtained assuming two possible cases, the CAT system only offers the best translation or the 5-best translations. In the latter case, the calculation of a given assessment metric was conducted considering that translation out of the five suggested translations that most minimises the corresponding error measure. As expected, there is a notable improvement when comparing 1-best to 5-best translation accuracy.

Analysing the results achieved with the *XRCE* corpus, it is observed that the KSR and KSMR rates for English-Spanish are substantially lower than those obtained in the rest of language pairs. A possible reason that explains these error rate discrepancies between English-Spanish with respect to English-German and English-French could be found in the test perplexity differences

Table 2: KSR and KSMR [%] for the XRCE (left) and EU (right) corpora using SFST as models.

	1-best		5-best	
XRCE	KSR	KSMR	KSR	KSMR
EnEs	13.00	21.83	11.22	19.19
EnDe	30.56	45.72	27.43	41.78
EnFr	30.19	43.81	27.29	40.10

	1-best		5-best	
EU	KSR	KSMR	KSR	KSMR
EnEs	21.32	33.04	19.30	29.94
EnDe	23.42	35.85	21.47	32.98
EnFr	19.51	30.08	17.50	27.04

shown in Table 1. The Spanish test perplexity is significantly lower than that of the rest of languages and this fact is transformed into better translation results.

This rationale is compatible with the results obtained for the *EU* corpus. In these results, the English-Spanish experiment exhibits similar error rates to those of the English-French pairs, but somewhat better than those of the English/German pairs. This same tendency is followed by the perplexity values appearing in Table 1. As observed, the German language seems to be more complex than the other languages and this is reflected in Table 2.

The figures of Table 2 clearly manifest a productivity gain when using the CAT system presented in this work. For example in the *XRCE* corpus, using five suggestions and translating from English into Spanish, the user would only need to perform 11.22% of the key-stroke interactions that would be required without this CAT system. On the other hand, the KSR results for the English-French and English-German experiments are 30.19% and 30.56%, respectively. Even in these cases, the number of key-stroke interactions is one third of that that would entail translating the same test set without a CAT system. If we consider the mouse interaction in the CAT evaluation, we can observe a 50% increment in the interaction rates, key strokes plus mouse actions, for most of the language pairs in both corpora. These figures reflect the fact that the productivy gain that CAT systems would theoretically provide is somewhat reduced by the interaction scheme that CAT systems required.

In the *EU* corpus, the best KSR results were obtained for the English-French experiment, followed by the English-Spanish results and, finally, the worst results were achieved for English-German. Despite the important difference in size between *XRCE* and *EU*, the results are similar and for some language pairs even lower in the *EU* corpus. As previously mentioned, the perplexity figures of both corpora partially explain these results. For instance, the English-French and English-German experiments present lower perplexity figures and better results in the *EU* corpus than in the *XRCE* corpus.

5 Conclusions and future work

In the present work, SFSTs have been revisited and applied to CAT. In this case, SFSTs that are easily learnt from parallel corpora were inferred by the GIATI technique, which was briefly reviewed. Moreover, the concept of interactive search has been introduced in this paper along with some well-known techniques, i.e. error-correcting parsing and n-best paths, that allow the calculation of the suffix translation that better completes the prefix written by the user. It is fundamental to remember that usability and response-time are vital features for CAT systems. CAT systems need to provide translation suffixes after each user interaction and this imposes the requirement of very efficient algorithms to solve the search problem.

The capability of SFSTs to suggest translation suffixes that aid a human translator to increase his or her productivity in a CAT framework should not be neglected. The results presented on two different corpora support the advantage of incorporating MT techniques into the CAT process to reduce human translator effort without sacrificing the high quality of the translations.

Given the relatively high possitioning effort (MAR) observed in the experiments, it seems worth investigating interaction modalities which are alternative or complementary to the traditional keyboard and mouse. In this respect, the use of speech interaction has been considered in (Vidal, Casacuberta, Rodríguez, Civera, & Martínez, 2006), with encouraging results.

Finally, the incorporation of confidence measures (Ueffing & Ney, 2005) and other appealing techniques (Bender et al., 2005) into the interactive MT scenario are topics still to be explored in future research.

References

Amengual, J. C., et al.. (2000). The EuTrans-I speech translation system. *Machine Translation*, *15*, 75–103.

Atos Origin, Instituto Tecnológico de Informática, RWTH Aachen, RALI Laboratory, Celer Soluciones and Société Gamma and Xerox Research Centre Europe. (2001). *TransType2 - Computer Assisted Translation. Project Technical Annex.*

Bangalore, S., & Riccardi, G. (1995). A finite-state approach to machine translation. In *Proc. of NAACL '01* (pp. 1–8). Morristown, NJ, USA: Association for Computational Linguistics.

Bender, O., et al.. (2005). Comparison of Generation Strategies for Interactive Machine Translation. In *Proc. of EAMT'05* (pp. 33–40). Budapest (Hungary).

Brown, P. F., et al.. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, *19*(2), 263-312.

Casacuberta, F., & Higuera, C. de la. (2000). Computational complexity of problems on probabilistic grammars and transducers. In A. Oliveira (Ed.), *Grammatical inference: Algorithms and applications* (Vol. 1891, pp. 15–24). Springer-Verlag.

Casacuberta, F., & Vidal, E. (2004). Machine translation with inferred stochastic finite-state transducers. *Computational Linguistics*, *30*(2), 205–225.

Casacuberta, F., et al.. (2004). Some approaches to statistical and finite-state speech-to-speech translation. *Computer Speech and Language*, *18*, 25-47.

Civera, J., et al.. (2004a). From machine translation to computer assisted translation using finite-state models. In *Proc. of EMNLP04*. Barcelona.

Civera, J., et al.. (2004b). A syntactic pattern recognition approach to computer assisted translation. In A. Fred, T. Caelli, A. Campilho, R. P. Duin, & D. de Ridder (Eds.), *Advances in statistical, structural and syntactical pattern recognition* (pp. 207–215). Springer-Verlag.

Jiménez, V. M., & Marzal, A. (1999). Computing the k shortest paths: a new algorithm and an experimental comparison. In J. S. Vitter & C. D. Zaroliagis (Eds.), *Algorithm engineering* (Vol. 1668, p. 15-29). Springer-Verlag.

Knight, K., & Al-Onaizan, Y. (1998). Translation with finite-state devices. In E. H. D. Farwell, L. Gerber (Ed.), *Proc. of AMTA'98* (Vol. 1529, pp. 421–437). London, UK: Springer-Verlag.

Och, F. J., & Ney, H. (2000). Improved statistical alignment models. In *Proc. of acl'00* (pp. 440–447). Hong Kong, China.

Picó, D., & Casacuberta, F. (2001). Some statistical-estimation methods for stochastic finite-state transducers. *Machine Learning*, *44*, 121-142.

Ueffing, N., & Ney, H. (2005). Application of Word-Level Confidence Measures in Interactive Statistical Machine Translation. In *Proc. of EAMT'05* (pp. 262–270). Budapest (Hungary).

Vidal, E., Casacuberta, F., Rodríguez, L., Civera, J., & Martínez, C. (2006). Computer-assisted translation using speech recognition. *IEEE Transactions on Speech and Audio Processing*, In press.

Viterbi, A. (1967). Error bounds for convolutional codes and a asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, *13*, 260–269.