university of groningen

Prediction Tool Progress in Interactive Machine Translation Systems

A study on the evaluation and improvement of the auto completion tool in Computer Aided Translation Tools

Chara Tsoukala

December 2014

Master Thesis

Human-Machine Communication

University of Groningen, The Netherlands

Internal supervisor:
Dr. Jennifer Spenader (Department of Artificial Intelligence, University of Groningen, NL)

External supervisor:
Prof. Philipp Koehn (School of Informatics, University of Edinburgh, UK)

Keywords

Interactive Translation Prediction, Statistical Machine Translation, Interactive Machine Translation, Computer Aided Translation tools, Auto-completion, Translation Process Study

Abstract

Machine Translation systems are still far from generating error-free translations, and the output usually requires human post-editing (PE) in order to achieve high-quality translations. The interactive-predictive machine translation (IMT) framework (Foster et al, 1997) comes to assist, rather than replace, human translators, and it aims to increase translation speed by adding real time suggestions to the human translation process.

This study focuses on Interactive Translation Prediction (ITP), an IMT tool that assists human translators by attempting to predict (autocomplete) the text that the user is going to insert. The input modality is familiar to anyone who has used autocomplete features in text editors, mobile phones or search engines. In ITP, the completion suggestions are obtained by matching the user input to the search graph, which contains all possible translations of all the segments in the source text. The detection of parts of the source text already translated is not a trivial task, especially given that, according to Usability Engineering standards, the mechanism needs to be fast enough to work at "user-typing" speed (i.e. max. 0.1 seconds). A typical way of obtaining the best match, and therefore the most probable completion, is to find the path in the search graph that has the smallest edit distance to the user input, and at the same time the highest path score.

The first goal of this research is to extract features, other than the edit distance and the path score, which are important for increasing the prediction accuracy, while taking into consideration the speed constraints. For this task, we treated word prediction as a Machine Learning problem using Support Vector Machines as the classifier. For the baseline algorithm, we used the prediction tool developed for Caitra (Koehn, 2009b, p.6), which is a dynamic programming solution that computes the minimum cost to reach each node of the search graph, by matching fully or partially the given user prefix. We generated a dataset using i) a modified version of the baseline algorithm and ii) 1144 post-edited sentences from the first field trial study of CasMaCat, a Computer Aided Translation (CAT) tool, and we explored features such as: a) whether the last token of the user input was matched to the last token of the matched string (lastMatched); b) whether the last 2 tokens were matched (last2Matched); c) whether the last 3 tokens were matched (last3Matched); d) levensthein (leven) distance between the last token of the prefix and the matched string (in case it is the same word but in e.g. plural form); e) whether the user input was longer than the matched string; f) the number of deletions (del), g) the number of insertions (ins) and h) the number of mismatches (msm) needed to match the prefix. Of these features, lastMatched resulted in higher prediction accuracy. The word prediction accuracy of the simulated evaluation, using the 1144 PE sentences, increased by an absolute of 0,5% from the baseline (55.6% to 56,1%).

For the official evaluation of the prediction tool accuracy, and to test its usability, a user study with 6 non-professional translators took place. The participants were asked to translate a number of sentences (newspaper corpus) from English to Spanish using two modes, PE and ITP; in the first mode, the participants were presented with the initial MT output that they had to post-edit, whereas in the second mode they saw the three next tokens of the translation suggestion in a floating box close to the caret position, i.e. the

edit box where they were typing their translation. During the whole study, the User Activity Data (UAD) was measured. After the completion of their translation sessions, the participants were asked to complete a quick survey in order to give feedback on the prediction tool. The hypothesis was that the participants would be in favour of ITP, as the tool constantly updates the suggestions, thus leading to completions closer to the user's needs. And it also requires less typing effort on their part**.** Indeed, questionnaire results show that the editors were strongly in favour of the interactive tool, but the logs do not show an increase in translation speed. On the contrary, in some cases the completion time is slightly lower in the case of PE. This implies that the interaction tool is not ready to replace the regular post-editing workflow in the translation industry yet. Nevertheless, given that the user satisfaction is high, it is worth further investigating a potential increase in accuracy, along with the optimal visualization option for interactive translation, such as the number of suggestions presented to the user (only the best or a list of suggestions in a drop-down menu), the number of tokens (the three first as in this study or more/less) and the place where the suggestions are displayed (directly in the editing box or externally as in this study, so as not to interfere with the translator's working space).

Contents

List of Acronyms

Acronyms

- CAT: Computer-Aided Translation
- FS: From Scratch
- IMT: Interactive Machine Translation
- ITP: Interactive Translation Prediction
- MT: Machine Translation
- ML: Machine Learning
- PE: Post Editing
- SMT: Statistical Machine Translation
- SVM: Support Vector Machines
- TAP: Think Aloud Protocol
- TM: Translation Memory
- UAD: User Activity Data
- WP: Word Prediction

Chapter 1.
Introduction

1.1 Problem description

During the past years, translation needs have increased dramatically due to globalization. Companies have the chance to expand their business in foreign markets more easily, but in order to do so they need to make sure that they can address their clients in their own language. Furthermore, institutions and political bodies such as the European Union have increased the need for translations, as the documents that affect all European partners need to be available in all official languages. For example, until recently the European Parliament[1] used to translate its proceedings into all 24 official languages of the European Union, something that is now done only upon demand due to its high cost.

In order to reduce translation costs and increase speed, Machine Translation (MT) could be used to provide automatically translated output. As Franz Och[2] stated in googleblog.blogspot.com (Och, 2012) while celebrating the achievements of Google Translate, "In a given day we translate roughly as much text as you'd find in 1 million books. To put it another way: what all the professional human translators in the world produce in a year, our system translates in roughly a single day".

However, despite important advances obtained so far in the field of Statistical Machine Translation (SMT), current MT systems are still not able to produce ready-to-use texts (Callison-Burch et al., 2007, Callison-Burch et al., 2008). Human post-editing (PE) of MT output is typically needed to achieve high-quality translations. Nevertheless, with PE, MT doesn't benefit (learn) from the user edits, and as a result the translators don't get the maximum assistance.

One way to use existing MT systems efficiently is to interactively combine them with the skills of a human translator, the so-called Interactive Machine Translation (IMT) paradigm (Foster et al, 1997) that we are going to focus on in this study. More specifically, we are going to focus on the prediction model that interactively suggests translations to the human translator by attempting to autocomplete their sentences based on the partial translation they have typed already (detailed information is given in the following sections, and Section 2.5 in particular).

This approach can be improved further by integrating the human knowledge into the Machine Translation system as well. This is done by using Adaptive Incremental Learning, like Online and Active Learning (e.g. Alabau et.al 2014) to re-estimate the parameters of the SMT model with the new translations (post-editions) that were generated and validated by the user (Ortiz-Martınez et al., 2010). By adapting the model, the SMT system is able to learn from the translation edits of the user and to prevent the

[1] http://www.europarl.europa.eu, accessed July 19, 2014

[2] At that point, Franz Och was the leading scientist at Google's MT group

repetition of errors in future MT output. This way, the system learns the user's preferences and the user benefits as well from the updated MT quality, thus leading to a virtuous cycle.

In this study we are focusing on the prediction model, which can be used independently or in combination with the adaptive models.

1.2 Current work

Interactive Machine Translation can be considered a special type of Computer-Aided, or Assisted, Translation (CAT) (Isabelle and Church, 1991). CAT is a form of human translation in which the translators are using software with tools, such as the ones that we will describe on Section 2.3, to assist themselves. In the past decade, Machine Translation and MT-based tools have been integrated into CAT tools. Examples of freely available CAT tools using simple Post Editing of MT output are the Google Translator Toolkit (Galvez and Bhansali, 2009) and the WikiBabel project (Kumaran et al, 2008).

In more advanced and recent CAT tools, human translation and MT are integrated tighter into the translation process. One way of achieving this is with the use of a prediction model that interactively suggests translations to the human translators by attempting to autocomplete based on their previous translation decisions, i.e. the partial translation they have typed already.

An example of the prediction model (auto completion) is given on Figure 1.1.

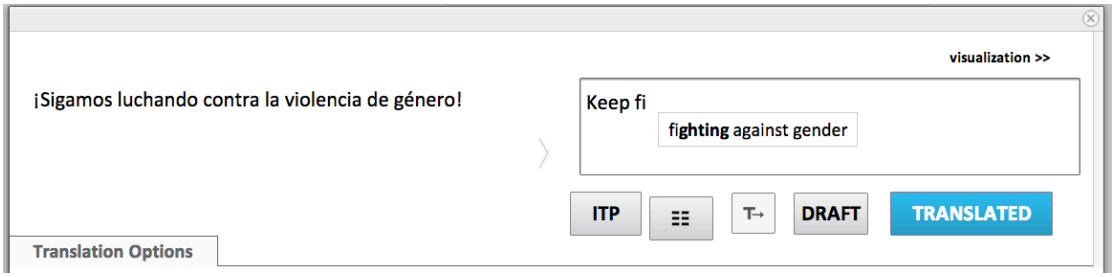


Figure 1.1.: Example of the auto completion tool within CasMaCat[1], an open source web based CAT tool (Appendix A). The autocompletion is displayed in bold, and it can either be a full word or a partial completion, as in this example (user input:"Keep fi", completion: "ghting against gender")

If the user doesn't like the translation suggestion, instead of accepting it he can keep typing, and new suggestions are generated. Examples of such tools are the projects

[1] http://www.casmacat.eu, accessed February 19, 2014

TransType (Langlais et al., 2000), Caitra (Koehn, 2009), and the TT2 project from Barrachina et al (2009).

In this work here, we are going to use the interface of one of the latest CAT tools released, the EU project CasMaCat (Cognitive Analysis and Statistical Methods for Advanced Computer Aided Translation[1]). As a baseline for the prediction tool, we are adapting the algorithm used in Caitra (Koehn, 2009b), which is the only known published algorithm for interactive auto completion.

Systems like the ones mentioned above are search-graph-based, because, in order to find the best completion of the human translation, they use the search graph generated along with the initial translation. More specifically, if the translator starts typing a translation that does not match the best MT suggestion, the interactive prediction tool quickly computes an error tolerant match in the search graph, and uses this as a starting point for the rest of the completions of the given sentence, until the translator diverges again from the suggestion etc. The best approximate match of the partial translation (a.k.a. the user input, or suffix) in the search graph is computed by finding the path that has the highest score and the smallest number of edits, a.k.a edit distance. The "completion prediction" is simply the most probable continuation of the path matched. A detailed description of the IMT process is given in Section 2.0.

1.3 Research question and objectives

The goal of this current work is to evaluate the usability of the Interactive Prediction tool, and to attempt to improve its accuracy by exploring additional features, namely:

a) whether the last token of the user input was matched to the last token of the matched string (lastMatched)

b) whether the last 2 tokens were matched (last2Matched)

c) whether the last 3 tokens were matched (last3Matched)

d) word level levensthein (leven) distance between the last token of the prefix and the matched string (in case it is the same word but in e.g. plural form)

e) the number of deletions (del),

f) the number of insertions (ins) and

g) the number of mismatches/ substitutions (msm) needed to match the prefix

h) whether the user input was longer than the matched string

These features, along with the path score and the edit distance, were manually selected as candidate features that can contribute to a successful generation of an accurate prediction. The hypothesis behind the individual number of deletions, insertions and substitutions, are that, for the same edit distance, deletions and insertions are more harmful than substitutions.

The purpose of the feature extraction task is to eventually lead to an improvement of the prediction algorithm and, in turn, current Interactive translation prediction (ITP) systems. The features are extracted using Machine Learning techniques (Support Vector Machines, SVM), and are evaluated against field trial datasets.

Last but not least, it has to be kept in mind that the goal of IMT systems is not to replace human translators, but to assist them by accelerating their work and improving their translation quality. Therefore, IMT systems need to take into account results from Usability Engineering that draws on work from human cognition, and make sure that the users, which in this case are the human translators, are indeed helped and not confused by the various IMT tools. We address these issues with a human evaluation of the prediction tool, using a web-based interface.

Chapter 2.
Theoretical background

2.1 Machine Translation

The most common and successful method of Machine Translation used currently is Statistical Machine Translation (SMT). Other methods are: i) Rule-based, ii) Example-based and iii) Hybrid MT (e.g. statistics guided by rules).

SMT has advanced greatly during the last decades. However, even though researchers try to build grammar-based translation models that take into account the linguistic features of language, the most popular models are still phrase-based.

Briefly, in phrase-based models, the source text (input) is segmented into text chunks, which may or may not correspond to linguistic phrases (e.g. noun phrases, verb phrases, and prepositional phrases). Each text chunk is translated and may be reordered (depending on the language pairs, i.e. the morphology of the source and target languages), and the final output is constructed with the help of a language model. The *language model* is responsible for the fluency and well-formedness of the output and it derives statistically from monolingual corpora of the target language. It is simply the probability of seeing a given sequence of words in the target language.

On the other hand, the *translation model* derives from parallel corpora (i.e. aligned texts that are available in two or more languages). The translation model is the probability of translating a phrase (from, e.g. English) into a certain phrase (in e.g. Spanish). Generally, the quality of the SMT output increases when more parallel corpora (a.k.a. the training corpus) is available; in fact, in order to build a reasonably fluent MT system, a few million parallel sentences (from the two language pairs) need to be used as a training corpus.

The translations of the individual phrases (text chunks) are called translation options. Typically, during the decoding of a source text, up to 20 translations for each text chunk are considered (Koehn, 2010).

The large number of the translation options and their even larger combination possibilities create a very large search space, which is costly to explore exhaustively. For this reason, heuristic algorithms are used in order to find the best translation. During the heuristic search, a search graph is constructed, which can be used later to generate the n-best translations that are needed in the Interactive Machine Translation process.

2.1.1 Machine Translation quality

As mentioned above, despite important advances obtained so far in the field of MT, current SMT systems, even the best ones, are still far from being able to produce high-quality texts.

A typical way to evaluate the MT quality is by using the BLEU (Bilingual Evaluation Understudy) score (Papineni K. et al, 2002), which is an automatic metric of evaluation of MT output given a (human) reference. BLEU uses a modified form of precision, and what it roughly does is to calculate the overlapped ngrams between the hypothesis (MT

output) and the reference (the good quality human translation). BLEU scores range from 0 to 1 (or 0-100%), and there is a correlation to human evaluation.

Two equally interesting automatic metrics are NIST and TER. NIST is based on BLEU, but it also calculates how informative a particular n-gram is by adding more weight to rare correct ngrams (Doddigton, 2002). TER (Translation Error Rate) (Snover et al., 2006) calculates the number of edits required to change the hypothesis (MT) translation into a reference translation by inserting, deleting, and substituting single words.

In Callison-Burch et al. (Callison-Burch et al., 2007), the authors evaluated the translation quality of several Machine Translation systems that participated in the WMT07 shared translation task[1] for 8 diverse language pairs: French-English, German-English, Spanish-English, Czech-English and vice-versa. The evaluation was done both with Human and automatic evaluation, with more emphasis given on the Human evaluation. The automatic metrics are the ones we mentioned above (BLEU, NIST, TER), plus 8 more. The human evaluation was done on a five point scale ranking that represents fluency and adequacy of the translation. The scales were developed for the annual NIST Machine Translation Evaluation Workshop by the Linguistics Data Consortium (LDC, 2005). Among all the systems that were evaluated, the best one for e.g. English-Spanish had BLEU score of 32.40, which is too low for publishable quality. Just as a reference, an average BLEU score when comparing two human translations (i.e. two different human translations used one as a reference and the other as a hypothesis) is 67.8 (Snover, 2006). The following year (Callison-Burch et al., 2008) the evaluation was repeated with new systems, and a new language pair was added on top of the others, namely English to and from Hungrarian, with similar conclusions.

In order to make it clearer why the MT quality is not good enough to be published without post-edition, we also demonstrate a simple example from our user study (Section 5). We compare here the MT output of two different state of the art SMT engines, Google translate[2] and a Moses[3] system that was originally built by the University of Edinburgh (Koehn and Haddow, 2012) for the translation task of the WMT12 evaluation campaign[4] which compares the output quality of several MT systems. According to the results of WMT12 (Callison-Burch et al., 2012), Google translate (ONLINE-B in the referenced paper) and Moses (UEDIN, respectively) are of comparable quality, with ONLINE-B ranked slightly higher (Callison-Burch et al., 2012, p. 17, Table 4)

---

[1] http://www.statmt.org/wmt07/ Second Workshop on Statistical Machine Translation. Accessed April 19, 2014

[2] https://translate.google.com/ Accessed July 19, 2014

[3] Moses (http://www.statmt.org/moses/, accessed July 19, 2014) is an Open Source Statistical Machine Translation engine that can be used to train models of textual translation from any source language to a target one, given the adequate resources (parallel corpus)

[4] http://www.statmt.org/wmt12/ Seventh Workshop on Statistical Machine Translation in Quebec, Canada. Accessed April 19, 2014

The example we used is for the English source sentence “Liars do look you in the eye.” (Appendix B, Segment 12). The output of the systems is the following:

Google (es): **Liars** te miran a los ojos.
Moses (es): Mentirosos **no les** mir**e**n a los ojos.

In the first example (Google), the word “Liars” was left completely untranslated. The statistical model seems to interpret “Liars” as a proper name, perhaps because it is in subject position and capitalized, deciding that the probability of it being proper noun is higher than a bare plural common noun. Therefore, its original form was preserved. It is interesting to note that if we lowercase “Liars”, then the MT output is correct (“mentirosos te miran a los ojos”), and the same happens if we add negation to the source (“Liars do not look you in the eyes” is translated as “Los mentirosos no te miran a los ojos”, which is also perfect).

In the second example (Moses), the word liars was correctly translated into *mentirosos*, but an erroneous negation was added to the translation, thus changing the meaning completely (the back translation is “Liars do *not* look you in the eyes”. Furthermore, Moses chose the subjunctive form of the verb *“mirar”* (look) “miren” instead of the indicative form (“miran”), which would be correct in this case.

Last but not least, Google chose an informal tone (“te”) whereas Moses the formal one (“les”). It is difficult to say which tone is the correct one, as it depends heavily on the context and the tone of the rest of the document. This is a simple example that demonstrates the variability of translation, which makes MT difficult to evaluate automatically. As we will demonstrate in the following Section (2.2.), human translation depends highly on the translator’s experience, style and background.

In conclusion, SMT can be relatively successful in helping humans get the gist of a foreign text, but there is no doubt that translation quality from qualified translators is more advanced than automatic translation. Therefore, for a publication-quality translation of official reports (such as the proceedings of the European Parliament mentioned on Section 1.1), books, web sites, movie subtitles and so on, Machine Translation can be used only as a supportive tool for human translators in the form of post-editing and Interactive Machine Translation.

2.2 Human Translation

Professional translators produce high-quality and accurate translations. However, their main drawback is their high cost, in terms of both money and time. Given the increase in translation needs due to globalization, the time constraints of the human translation is a major problem.

Furthermore, a professional translator needs to have two sets of skills in order to translate a document. The first skill is, of course, the *language skill*, which indicates the ability of fully understanding the source language, and the ability to produce fluent texts in the target language. The second skill a translator needs to possess is the *domain*

knowledge, namely the ability to understand a very specialized technical document. Both skills may be difficult to find, especially depending on the language pair or the domain.

Human Translation is also performed in non-professional environments by volunteer translators who are usually less qualified. For example, WikiPedia articles are translated by volunteer translators using the WikiBabel project (Kumaran et al, 2008), and so are movie subtitles[1] and TED talks[2], or news articles from around the world[3]. A user study of the CAT tool Caitra where non professional translators participated, has shown that the users were particularly able to increase their productivity and quality of work when assisted by machine translation or other translation tools (Koehn and Haddow, 2009).

One important feature of the human translation is its variability. This applies mainly to longer translation segments, but it has to be taken into consideration that a source sentence does not have only one correct translation. For example, from our user study that will be discussed on Section 6.0, the source English sentence:

> Source: "Now granted, many of those are white lies."

Was translated by 6 different native speakers of Spanish as:

1. Ahora es cierto, muchas de esas son mentiras piadosas.
2. Ahora de acuerdo, muchas de esas son mentiras piadosas.
3. Ahora ya aceptado, muchas de esas son mentiras piadosas.
4. Se da por sentado que muchas de esas mentiras son piadosas.
5. Ahora beneficiado, muchas de esas son mentiras piadosas.
6. Si bien es verdad que muchas de esas mentiras son mentiras piadosas.

This simple example clearly demonstrates the variability of human translation, which depends heavily on the translator's style and experience. It has to be kept in mind that this also makes the evaluation of the Machine Translated output a difficult task.

Even though not all human translators embrace the idea of working explicitly with Machine Translated output, most of them do use a number of computer tools (MT-based or not) to facilitate their work.

2.3 Translation Tools

The use of computers over the years in general, and CAT tools in particular, has increased the productivity of human translators and therefore lowered their cost. A number of translation tools have been able to facilitate their work and increase the quality of their translations. Example of translation tools are (Desilets, 2009):

- **Spell checkers**
- **Grammar checkers**

---

[1] http://www.opensubtitles.com, accessed July 19, 2014

[2] http://www.ted.com/translate, accessed July 19, 2014

[3] http://globalvoicesonline.org/about/, accessed October 27, 2014

- **Online dictionaries and thesauri**
- **Terminology databases:** contains terminology from different domains, such as medicine, law, computer science and so on
- **Translation memory (TM):** Translation Memories are databases of already (human) translated and approved phrases, which are queried for exact or fuzzy matching. By fuzzy matching we mean translated sentences similar to the one that the human translator is working on. These suggestions are then proposed to the user who can accept them fully or post-edit them. Unfortunately, TMs can only successfully match a small percentage of the total document, so it is not sufficient for assisting a complete translation.
  Translation Memories can be used both online and offline. It is also typical for translators to use TMs from segments they have previously translated themselves, especially if they are working on a specific domain. This also allows them to work offline. Example of a popular TM database is the database of SDL Trados[1].
- **Monolingual and bilingual concordances:** In concordance tools, words are shown in context, as used in actual texts. The bilingual concordance also shows the translation of the word, and it helps in meaning disambiguation (it needs to be kept in mind that a word can have multiple translations depending on the context).

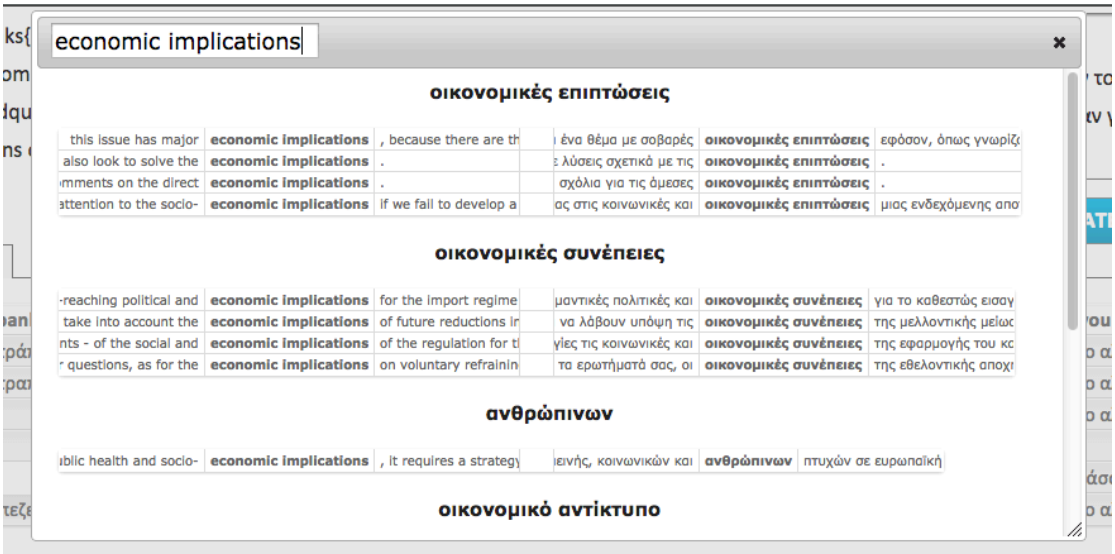


Figure 2.1: An example of the biconcordance tool in CasMaCat. The input (here: the English phrase "economic implications") can either be a single word or a full phrase. The tool then suggests several translations in the target language (here: Greek) sorted by their score (probability), and it also outputs the source and target phrases where the phrase occurs

Professional translators have only recently started using Machine Translation for post-editing instead of, or in combination with, Translation Memories. Therefore, there is rich

[1] http://www.trados.com, accessed July 19, 2014

potential for improvements and entirely new tools (Koehn, 2010). Already, some of the above-mentioned tools, like the biconcordancer (Figure 2.1), can be based on Machine Translation.

2.4 IMT and prediction using Search graphs

As mentioned in Section 1.2, typical implementations of IMT systems are based on the generation of word translation graphs (aka search graphs). While a given sentence is being translated, the IMT system makes use of the search graph that is generated for that source sentence, in order to complete the input given by the human translator. More specifically, the system finds the best path in the search graph which matches the user input, by selecting the path that has the highest score and the smallest edit distance.

Search-graph-based IMT systems are popular because of their efficiency in terms of time cost per interaction. This is due to the fact that the search graph is generated only once, along with the initial (best) translation, namely at the beginning of the interactive translation process of a given source sentence. Therefore, the completions (predictions) required in IMT can be obtained by processing only the search graph, without further involving the Machine Translation engines, something that would be very costly.

However, a typical problem in IMT is that the user may insert a phrase that cannot be matched in the search graph. In this case, the completion cannot be generated successfully, because the system is unable to predict translations that are compatible with the input given by the user.

In the IMT systems that rely on search graphs to generate the completion, the common procedure to deal with this problem is to perform an error-tolerant search of the user input in the graph. An example of this is Phrase Edit Distance, an error-tolerant search that uses the well-known Levenshtein distance (Levenshtein, 1966) in order to find a match in the search graph that is most similar to the user input. The edit distance technique is crucial to generate the best matching completion of the user's input. This current work explores the possibility of additional features that, along with the path score and the edit distance, can account for an accurate prediction.

For the clarification of the IMT process, a formal description of the IMT framework is given in the Section below.

2.5 SMT and IMT Frameworks

The IMT framework is an alternative to fully automated MT systems. In the case of IMT, the Machine Translation system that assists the human translator attempts to predict (and autocomplete) the text that the user is going to type. This is done taking into account the user's previous translation choices. Whenever such prediction is wrong and the user provides feedback to the system by changing the completion, a new prediction is performed, this time including the user's most recent feedback (which is the partial translation) as the input of the prediction tool. The process is repeated until the human translator is satisfied with the generated translation.

Specifically, when the users start translating a text, they are given a Machine Translated output which they are requested to post edit in case it contains errors or it lacks fluency. From the moment they start editing, the IMT process starts. At each interaction of the IMT process, the IMT system uses the search graph to generate a new translation of the source sentence, which can be partially or completely accepted and corrected by the user of the IMT system. Whenever new edits are made, each partially corrected text segment (also known as a prefix) is used as an input to the IMT system in order to generate translation suggestions that match best the user's expectations.

More formally, the IMT framework can be seen as an extension of the SMT framework, which is described below:

In phrase based translations, a document is translated according to the highest probability distribution $\arg_y\max P(x|y)$ that a given string x in the target language (for example, Spanish) is the translation of a string y in the source language (e.g. English). Denoting y as the target translation string, and x as the source text, the fundamental equation of the statistical approach to MT is:

$$\hat{y} = \arg_y\max P(y|x) = \quad (2.1)$$

$$\arg_y\max P(x|y)P_{LM}(y) \quad (2.2)$$

where P(x|y) is the *translation model*, which models the correlation between the source and the target sentence, and $P_{LM}(y)$ is the *language model*, which represents the fluency and well-formedness of a candidate translation y.

More specifically, the translation model is the probability that the source string is the translation of the target string, and the language model $P_{LM}(y)$ is the probability of seeing this specific language string, or sequence of words, in the target language. Both are derived statistically from corpora. The translation model is created from parallel corpora, for example texts that have already been translated into one or more languages (e.g. texts that exist in both English and Spanish), and the language model is induced from a monolingual corpus in the target language.

As mentioned above, the SMT equation can be easily extended to describe the IMT scenario. In the IMT framework, we need to take into account that part of the target sentence has already been translated by the translator (namely, the user input or prefix). So Eq. 2.1 changes to include a prefix y_p that is given by the user, in order to find an extension $\hat{y}_s$:

$$\hat{y}_s = \arg_{y_s}\max \{ p(y_s \mid x , y_p) \} \quad (2.3)$$

according to the highest probability distribution of the suffix y_s.

By applying the Bayes rule, we arrive at the following expression:

$$\hat{y}_s = \arg_{ys}\max \{ p(y_s \mid y_p) \bullet p(x \mid y_p, y_s) \} \quad (2.4)$$

where the term $p(y_p)$ has been dropped since it does not depend on y_s.

Therefore, the search is restricted to those sentences that contain y_p as prefix.

An example of a typical IMT session, as described above, is illustrated in Figure 2.1

source sentence: I need to print my flight tickets
desired translation: Necesito imprimir los billetes de avión

iter.-0:	Necesito mi para imprimir billetes de avión
iter.-1:	Necesito i mprimir billetes de mi vuelo
iter.-2:	Necesito imprimir l os billetes de avión
accept	Necesito imprimir los billetes de avión

Figure 2.2: a typical IMT session where an English sentence is translated into Spanish. The desired translation is the translation that the user has in mind. At interaction-0, the system suggests a translation. At interaction-1, the user moves the cursor to accept the first word ("Necesito") and presses the 'i' key. At that point, the system suggests a new completion of the sentence with ("mprimir billetes de mi vuelo"). The next interaction is similar to interaction-1. In the final interaction, the user accepts the given translation.

More information on the typical approach to IMT and the baseline prediction algorithm in most IMT systems is given on Section 2.6.

2.6 Prediction tool algorithm in IMT systems

As mentioned before, typical Interactive Machine Translation systems (e.g. Langlais et al, 2000, Barrachina et al., 2009, Koehn, 2009) aim to predict the best matching continuation of the user input given the source language text and a partial translation. The biggest challenge of interactive translation prediction systems is to successfully

match what has been translated already, even if the user introduces words that have not been seen by the decoder.

Usually, the MT system returns the n best translations of a given phrase in a search graph (Figure 2.3), and the prediction is done by matching the user input against this search graph and outputting the remaining most probable path. As the translator makes edits, diverging from the initial suggestion, the prediction tool examines only the search graph instead of interacting with the MT decoder. This approach is considerably more efficient, as it saves a lot of processing time and can therefore lead to faster results. An alternative to this is to use force decoding, or prefix decoding. As the name implies, in this case the decoder is forced to produce a translation that matches the prefix (partial translation given), and then it's free to produce the rest of the translation. Green et al. follow this approach to IMT using Phrasal, an SMT toolkit (Green et al, 2014). However, this doesn't solve the problem of prediction failure when the user prefix contains words that have not been seen by the decoder. Plus, the decoder has to be very fast in order for forced decoding to be usable in IMT scenarios.

Speed is a major issue in this case, because the results have to be displayed on the user's screen in typing speed, so they should not take more than a few milliseconds to compute. In fact, according to standards in Usability Engineering and Response Times, "0.1 second is about the limit for having the user feel that the system is *reacting instantaneously*, meaning that no special feedback is necessary except to display the result" (Nielsen, 1993).

It has to be noted here that the fact that the prediction is heavily dependent on the decoder (in the means of a search graph) implies that the MT system quality is also very important for a successful prediction. The correlation between MT output quality and post editing effort has caught the attention of several researchers (e.g. Koponen et al. 2012, Koehn and Germann 2014). What is also interesting is that the notion of MT output quality is highly subjective (Koponen, 2012, Turchi et al. 2013). However, we are not going to focus on decoders with different output quality in this study, nor on the subjectivity of the evaluation. Instead, our goal is to extract general features that can help improve the accuracy of the prediction tool that uses the search graph produced by SMT decoders.

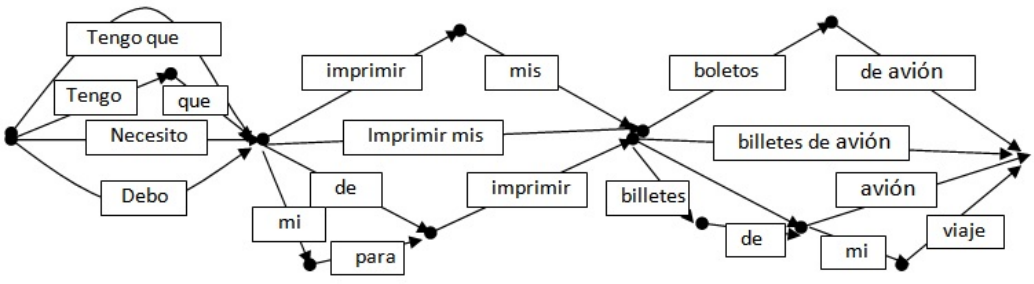


Figure 2.3: Example of a simplified search graph of n best translations in Spanish for the source English phrase "I need to print my flight tickets". Each vertex has an optimal path leading to it.

In phrase-based SMT models, each vertex in the graph (e.g. Figure 2.3) has an optimal path leading into it, which has to be matched by the user input, and an optimal path leading to a full translation. Therefore, the approximate search problem looks for the vertex that best matches the user input (prefix). The optimal path leaving the vertex is the sentence completion, or what we call here prediction or suggestion.

The baseline of the prediction tool used in this work is Caitra's (Koehn, 2009b, p.6) prediction algorithm (Figure 2.4). It is a dynamic programming solution that computes the minimum cost to reach each node of the search graph, by matching fully or partially the given user prefix. The algorithm uses string edit distance as primary objective, and path score in the search graph as secondary objective.

Input: user prefix u, search graph g
Output: best path p

allowable error $e = 0$
best path $p_i = \{\}$ for all error i
add backpointer (cost=0.0, error=0, toProcess=u) to start state
while best path $p_{e-1} == \{\}$ and error $e <$ length(p) **do**
  for all state $s \in g$ in topologically increasing order **do**
    for all backpointer b of state s **do**
      if b.error $== e$ **then**
        for all transition t from state s **do**
          compute string edit distance matrix for b.toProcess, t.phrase
          for all matches m in matrix that consumed all of b.toProcess **do**
            new cost $c_n = s$.cost + t.cost + t.toState.forwardCost
            new error $e_n = s$.error + m.error
            if $c_n < p_{e_n}$.cost **then** set this as p_{e_n}
          end for
          for all matches m in the matrix that consumed all of t.phrase **do**
            reached new state $s_n = t$.toState
            create new backpointer b_n
            b_n.cost=s.cost+t.cost
            b_n.error=s.error+t.error
            b_n.toProcess=s.toProcess-t.phrase
            b_c = current backpointer for state s_n at prefix pos. b_n.toProcess
            if b_c not defined **or** b_n.error $<$ b_c.error **or** b_n.error $==$ b_c.error
            and b_n.cost $<$ b_c.cost **then**
              make b_n new backpointer for state s_n, pos. b_n.toProcess
            end if
          end for
        end for
      end if
    end for
  end for
end while
best path $p = p_e$

Figure 2.4: Taken from Koehn, 2009b, p.6. This algorithm finds the best match for a prefix in a given search graph

The worst-case complexity of the baseline algorithm is linear in the number of states and quadratic in the length of the user input (given finite limits on state fan-out and phrase lengths), but in practice it is much faster (Koehn, 2009b)

The edit distance (Figure 2.4, line 9) and model cost (Figure 2.4, line 18) need to be computed for all substrings of the prefix, as the user may type something that appears at the beginning, the middle, or the end of a sentence.

Edit distance is the minimum string edit distance, which is the number of edit operations, namely: insertions, deletions and substitutions needed to turn one sequence into the other. For example, if the user has typed "Neces" as a partial Spanish translation of the source English phrase "I need", then if we wanted to match the user input to the following words we'd need the according number of edits:

Necesito	3 ins	**Tengo**	4subs
Debo	4 subs	**Tengo que**	4subs+4ins (8edits)

Therefore "Necesito" has the minimum edit distance, meaning that it requires less edits.

The match to the search graph is done iteratively, and every time there is a matching error (in the string edit distance), the number of allowable edits is increasing.

In conclusion, for the time being the phrase edit distance and the model cost (path score) are the only features used to compute the best match of the user input in the search graph. Therefore, there is much potential for improvements.

Chapter 3.
Feature extraction for the improvement of the prediction accuracy

By treating translation prediction (word completion) as a Machine Learning problem, we develop a classifier that is trained on human post-editing data. The challenge is to extract and test important features for increasing the accuracy of the prediction.

The first step is to determine the accuracy of the baseline algorithm using real field trial data (section 3.1). As a baseline we are taking the prediction algorithm that was proposed by (Koehn, 2009b, Figure 4) for Caitra. The algorithm uses string edit distance as primary objective, and path score in the search graph as secondary objective. When exploring the search graph, there are many possible prefix matches that, therefore, lead to different optimal path completions and predictions. The baseline algorithm looks for the highest probable path among the minimal edit distance matches in order to find the most probable prediction; our goal is to extract equally important features that can lead to successful prediction, and extend the baseline algorithm with those features.

After the initial evaluation, candidate features are extracted and tested using Support Vector Machines (Section 3.3). By modifying the baseline algorithm, given a user prefix and a search graph we can sample a set of alternative matches to the prefix $\{y_s^1, y_s^2, \ldots y_s^n\}$, for which we have a number of features such as:

1. $h_1(y_s)$ = string edit distance
2. $h_2(y_s)$ = log of the path probability
3. $h_3(y_s)$ = whether the last word of the prefix was matched in the graph

The full list of features is described in Section 3.2.1. Given this set of features, we define a linear model

$$\hat{y} = \arg_{ys}\max\{ \sum_i \lambda_i \, h_i(y_s) \}$$

The goal of the Machine Learning problem is to find the optimal weights ($\Lambda = \{\lambda 1, \ldots, \lambda n\}$) of the selected features. From our dataset of post-edits, we know the correct completion of each user prefix. Therefore, the extracted features can be used as supervised training data for a binary classifier.

Finally, the extended algorithm that includes features from section 3.2 is evaluated (section 3.4) using the same dataset as in section 3.1.

3.1 Evaluation (baseline)

Before starting with the feature extraction, a first evaluation of the performance of the prediction algorithm used in Caitra (Koehn, 2009b) needed to be made, in order to have a baseline. For this evaluation, we used the **FTD_2012_CasMaCat** dataset, which is the

post edited data of the first official Field Trial of CasMaCat[1] that took place in June and July 2012 in Madrid, Spain, at the offices of the translation service company "Celer Soluciones" that is a CasMaCat partner.

3.1.1 Dataset

FTD_2012_CasMaCat consists of newspaper articles; the newspaper corpus is taken from WMT12[2] (Callison-Burch et al. 2012) and it contains texts from CNN, Washington Post, LA Times, NY Times, Fox News and The Economist.

Five professional translators were asked to translate the above mentioned corpus from English (source language) to Spanish (target language) by either Post Editing the MT output (PE), or translating from scratch (FS). The final dataset consists of 1144 post edited sentences. Sentences that were translated from scratch (FS) were not included neither in the evaluation nor the main analysis, but we explore the data FS in Sections 3.1.3 and 3.2.2. The search graphs come from the Machine Translation engine used at the time of the Field Trial was a state of the art Moses[3] system that was originally built by the University of Edinburgh for the translation task of the WMT12 evaluation campaign (Koehn and Haddow, 2012). No extra pruning was used to discard nodes from the search graph that belong to paths with low score. Threshold pruning is important for speed reduction, but it can lead to an increased failure rate as it limits the coverage.

Interactive Translation Prediction (ITP) using the prediction tool was not tested in the first field trial of CasMaCat in Madrid. It has to be kept in mind that the lack of interactive translation data in the FTD_2012_CasMaCat dataset may falsely decrease the total accuracy in the automatic evaluation of the prediction tool that we are performing (both the baseline and the final algorithm). In the case of interactive translation, the translators might have accepted an equally correct alternative completion that would have been generated by the search graph, even if it wasn't their first choice. Hence, this evaluation can only show the 'floor' accuracy.

3.1.2 Method

For each of the 1144 post-edited sentences of the **FTD_2012_CasMaCat** dataset, we tested how often the predicted word matched the users' desired input (the target word they had already typed in the field trial). The initial match was against the MT output, and for every word there was a mismatch, a new prediction was generated, and the user's post edited output was then matched against the *new* translation prediction. These steps are meant to simulate the user's interactive translation process as described in Figure 2.2

---

[1] Cognitive Analysis and Statistical Methods for Advanced Computer Aided Translation, the CAT tool mentioned in Section 2 and Appendix A.

[2] http://www.statmt.org/wmt12/ Seventh Workshop on Statistical Machine Translation in Quebec, Canada. Accessed April 19, 2014

[3] http://www.statmt.org/moses/, accessed July 19, 2014

and more analytically on Section 2.6. We also added a timeout (10s) which is still long, but closer to a realistic scenario. The reasoning behind it is that, based on the Usability Engineering standards, we have to find a tradeoff between speed and prediction quality; we don't care about correct results that came from an exhaustive search, because they will be displayed too late for the user to accept.

The final accuracy score per sentence was the percentage of correctly predicted words against the total words. The final accuracy of the baseline is 55.55%.

3.1.3 Absolute maximum accuracy of word predictions

In order to ensure that the search graphs indeed contain the post edited words that had been chosen by the translators, we also measured how often the word that needed to be predicted (i.e. the word that the translator was going to type next) was included in the search graph. For this task the same dataset was used (**FTD_2012_CasMaCat**), but this time the translations that were created from scratch (FS) were included. The percentage of the existence of the desired word anywhere in the search graph was tested.

Out of a total of 61756 words tested, 91.97% of words existed in the search graph (67145 words out of 61756). When we tested only the Post Edited output (without data FS), the percentage went up to 93.57% (26343 out of 28153 words). Therefore, using the search graph to predict the user's translation is indeed good practice.

3.2 Word prediction as a Machine Learning problem

3.2.1. ML dataset

In order to create a dataset for the Machine Learning (ML) task, the **FTD_2012_CasMaCat** dataset was used once again. For each sentence and each token, the baseline prediction algorithm was modified so as to output *all* the possible matches to the prefix. Therefore, not only the "winning" prediction (which in the baseline is the one with the lowest edit distance and the highest path score), but all the alternative predictions were generated. At the same time, candidate features that could be used to increase the accuracy of prediction were included in the output. The full list of features is given in Table 3.1.

Using i) the tokens of the FTD_2012_CasMaCat post edited dataset and ii) the prediction tool, we collected data for each Word Prediction (WP). By WP, we define the times that the prediction tool was called in order to produce new suggestions for the user input; therefore, one post edited sentence may have multiple WPs, because whenever there is a mismatch between the user preference and the suggestion, a new suggestion is computed. In this case, WP and the number of tokens are the same, because in this experimental setup the prediction tool is called for all the words of the phrase. However, in a non-simulated setting (a usual interaction with the tool), the prediction tool only returns a new suggestion when there is a mismatch between the best suggestion string and the partial translation that the user has typed.

Table 3.1: Features used and their explanation

pathScore	the total path score, i.e. the score of the matched prefix (user input) and the suffix (completion). This feature is already used by the baseline algorithm
avgPathScore	the average path score (score/states). It is the normalized pathScore, used specifically in the ML part and not in the final prediction algorithm
sed	the search edit distance (sed) score, i.e. total number of edits (insertions, deletions, substitutions) needed
lastMatched	whether the last token of the user input was matched to the last token of the matched string. The hypothesis behind it is that it is more important for an accurate prediction to match the last word that the user typed than the words at the beginning of the user prefix.
last2Matched	Similar to lastMatched, but indicates whether the last 2 tokens were matched
last3Matched	Similarly, whether the last 3 tokens were matched
avgSed	the sed averaged by number of tokens of the matched prefix (normalized sed)
states	the number of states of the matched path (used for the normalization of sed)
largerMatched	whether the user input was larger than the matched string
leven	word level levensthein distance between the last token of the prefix and the matched string. It is similar to lastMatched, but this feature tries to catch the cases where the user's token is the same word as the last matched one, but in other form, e.g. plural
msm	Number of mismatches,
ins	number of insertions and
del	number of deletions that compose the sed. The hypothesis behind it is that a mismatch is less costly than an insertion or deletion.

The ML dataset that we constructed is ML_FTD12 (Field trial data 2012 for Machine Learning). From this dataset, a subset (ML_FTD12_100) was created that contained only the 100-best predictions (sorted by path score) per WP.

ML_FTD12 and its subset contain all the features of Table 3.1, plus:

1. The WP id (to enable manipulation of the data)
2. The word that should be predicted (i.e. the word that the user typed)

3. The matched string and the according prediction (in the form: matched string# prediction)
4. Only the matched string to the user prefix
5. The user prefix
6. The number of errors to reach the matching
7. Whether the prefix size was larger than the "matched" string length
8. Whether it was the winning prediction (only one for each WP)
9. Whether it was a correct prediction (first prediction token matching the word that should be predicted [2])

An example of the alternative paths considered for the generation of the ML dataset using the modified prediction tool is given here:

Source phrase: The cat is sick
MT output: El gato está enfermo
Reference (PE phrase): La gata está enferma

The prediction tool will be triggered even after the first reference token ("La"), because the Machine Translated text was wrong (it suggested "el gato", i.e. that the cat is masculine, whereas the translator had in mind a female cat-"la gata").

So, in our example the first input would be **la#gata**, where "la" is the token that the user typed, and "gata" the desired prediction. The output would be a list of the alternative paths, plus the above mentioned features.

Table 3.2: Alternative matched and prediction paths for the input string "la"

Matched string	Prediction
gata	está enfermo
se	está enfermo
al	gato está enfermo
los	gatos está enfermo
el	gato está enfermo
la	**gata** está enfermo
es	el gato enfermo

3.2.2 Oracle prediction

On Section 3.1.3 we talked about the absolute maximum accuracy, which is the percentage of the times that the user input was found in the search graph, which is 93.57% in our post-edited data and the search graphs of the decoder used in the field

trial. However, it is more interesting to check a more realistic maximum accuracy, namely the so-called "oracle" prediction. Starting from the baseline algorithm and the alternative phrases that were considered (e.g. Table 3.2), we want to see how often there was *at least one* accurate prediction in the **ML_FTD12_100** set of alternative matches, which contains the 100best alternative suggestions.

The final percentage (a.k.a. "oracle") is **71.15%** (8937 / 12560 unique WPs), which indicates the maximum accuracy possible for the current prediction algorithm. This means that even if we extract all features possible to increase accuracy, the maximum we can get based on this approach is 71.15%.

Even though it would not be a realistic scenario due to the speed constraints mentioned for Usability Engineering, we also tested the oracle prediction on the full dataset (**ML_FTD12**, so not only of the top 100 per WP). In this case, the percentage of at least one correct prediction in every set of predictions is 79.62% (10001 / 12560 unique WPs).

3.2.3 Feature exploration

In this Section the goal is to select candidate features from ML_FTD12_100. For this task, two different techniques were applied, both using the scikit-learn package for Python (Pedregosa et al., 2011).

3.2.3.1. k-best features

The first approach was to look up the k best features, using the scikit-learn's function SelectKbest, which performs univariate analysis and selects the k lowest p-values. In the dataset **ML_FTD12_100,** for k=3 and the results were:

1. average path score
2. search edit distance
3. last matched

which correlates well with what we would expect, as in the baseline algorithm emphasis is given on the path score and the edit distance. Therefore, matching the last word seems to be the next most important feature.

3.2.3.2 Exploring features using SVM

For the second approach we used the data extracted at Section 3.2.1 and a Support Vector Machine (SVM) classifier, more specifically the Radial Basis Function (RBF) kernel. For this task, we used an SVM implementation based on libsvm.

The dataset used is a subset of **ML_FTD12_100**. The difference is that we used only 2 lines for each WP: an accurate prediction, and an erroneous one. The goal is to compare these two classes, and be able to extract features that account for a good prediction.

The new dataset is **ML_FTD12_100_2SVM**. From this dataset, 28.93% of the WPs that contained no accurate prediction at all were excluded, because they would increase the number of false negative classifications.

The dataset was split into training (13172 lines, so 6586 WPs), development (1650 lines, 825 WPs) and test (1614 lines, 807 WPs) set. The results of the classification of the test set can be seen on Table 3.3.

It is important to note that the numbers indicate the percentage of correct classifications performed by the SVM classifier; therefore, the numbers cannot be compared directly to the ones mentioned in the previous sections, because those are based on the accuracy of the baseline algorithm tested on field trial data (FTD_2012_CasMaCat) and not on the classification

Table 3.3: Some of the features tested, and their classification using SVM on the test set. From the table it derives that the matching of the last word of the prefix (lastMatched) leads to more correct classification, whereas the levenshtein distance (leven) and the number of deletions (del) do not add to the model.

	Correct	False positives	False negatives
All features	1296 (80.30%)	156 (9.66%)	162 (10.04%)
sed + lastMatched	1341 (83.09%)	135 (8.36%)	138 (8.55%)
sed+lastMatched+leven	1340 (83.02%)	135 (8.36%)	139 (8.62%)
sed+lastMatched+msm	1342 (83.15%)	134 (8.30%)	138 (8.55%)
sed+lastMatched+msm+ins	1345 (83.33%)	161 (9.98%)	108 (6.69%)
sed+lastMatched+msm+ins+del	1344 (83.27%)	162 (10.04%)	108 (6.69%)

3.3 Evaluation of the extended model

In this part of the task, the prediction algorithm was extended by two of the best classifiers from Table 3.3: i) lastMatched (sed+lastMatched) and ii) number of mismatches and insertions (sed+lastMatched+msm+ins). Both algorithms were evaluated against the first year trial data. As mentioned in section 1.1, the accuracy of the original algorithm among the 1144 post edited sentences of the FTD_2012_CasMaCat dataset is 55.55%. However, using the classifier to decide on the optimal path with the same long timeout (10s), the accuracy dropped to 55.28% for sed+lastMatched and 55.12% for sed+lastMatched+msm+ins, because of the added computation cost.

The results of the ML classifier may have been disencouraging, but it led to a hands-on approach to the prediction algorithm by extending the actual baseline to include one of the winning features, lastMatched, without the help of the classifier. The reason for this

selection (lastMatched) is that it is one of the best selected features from the feature extraction task, and, at the same time, it is intuitive that more emphasis needs to be given on the last word that the user typed, as it gives more relevant clues as what needs to be typed next; it is therefore more crucial to match what the user has typed last than the translation that belongs to the beginning of a certain sentence.

The extended prediction code was evaluated in the same way as in the section 4.1, using the FTD_2012_CasMaCat data. With the extended feature (lastMatched), the accuracy increased by an absolute of 0.5%, reaching 56.1%, therefore leading to the conclusion that a hands-on approach to the algorithm is more fruitful than treating translation prediction as a Machine Learning problem.

3.4 Conclusion and Future work

From the oracle prediction and the fact that 93.57% of the post edited (PE) words exist in the search graph, it derives that there is indeed space for improvement in the accuracy of the prediction. However, the Machine Learning approach does not seem to add useful information to the model, mainly due to its high-added computation cost.

An alternative and more fruitful approach is to focus mostly on features related to the last words of the user prefix, and to apply them directly on the prediction algorithm, while evaluating the changes in performance as measured in accuracy and speed (Section 3.4.1). In the months following the work described in this Section (3), Koehn et al (Koehn et al, 2014) followed this approach with very good results. By pruning the search graph, having less strict matching criteria and emphasising on the last word that the user typed, the word prediction accuracy increased by an absolute of 5.4% (from 56.1% to 60.5%).

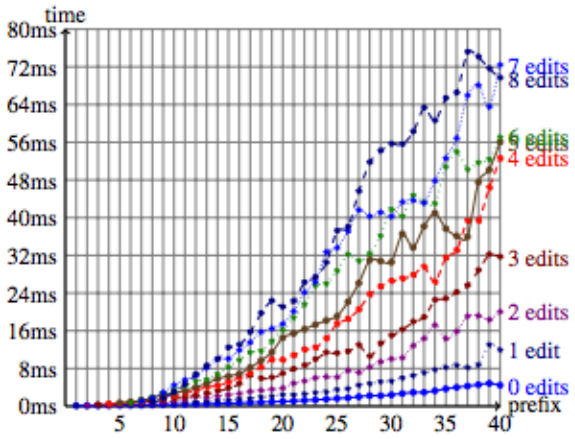


Figure 3.1: Taken from (Koehn et al, 2014, p. 2, Figure 1). It displays the processing time in ms against the length of the user prefix and the string edit distance between the user prefix and the search graph

It is also interesting to note the analysis of the processing time of the baseline algorithm, according to the string edit distance between the user prefix and the search graph. The plot on Figure 3.1 is taken from (Koehn et al, 2014, p. 2) and displays the processing time in ms against the length of the user prefix (up to 40 tokens), and the string edit distance between the user prefix and the search graph.

The graph clearly demonstrates that, especially for sentences that contain more than 10 tokens, an increased number of edits is very costly. However, as we mentioned in Section 2.6, according to Usability Engineering standards, the system should respond in maximum 100 milliseconds (0.1 second) in order to have the user feel that the system is *reacting instantaneously* (Nielsen, 1993). For that reason, in Koehn et al. (Koehn et al, 2014) the algorithm aborts when this time is exceeded, something that happens frequently for sentences larger than 20 tokens, as Figure 3.2 demonstrates.

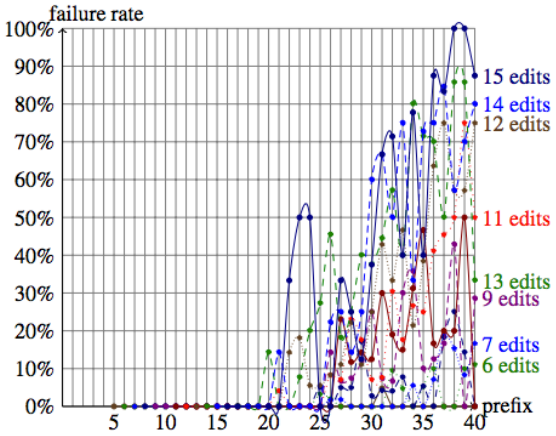


Figure 3.2: Taken from (Koehn et al, 2014, p. 2, Figure 1). The plot displays the ratio of failed predictions (due to the 100 ms limit) to the number of edits

In (Koehn et al, 2014) the experimental setup is the same as here. A simulated setting, as described in Section 3.1.2, took place instead of a user study. The FTD_2012_CasMaCat dataset was once again used, as well as the search graphs generated from the competitive English-Spanish MT system (Koehn and Haddow, 2012) for the first field trial of CasMaCat, with the difference that threshold pruning (as mentioned in Section 3.1.1) was applied to the search graphs, in an attempt to balance the trade-off between speed and accuracy. Threshold pruning means that nodes that belong to a path that is worse than the best path by a specific threshold are removed from the search graph. Table 3.4.1 shows the impact of threshold pruning to the accuracy and failure rate of the algorithm (i.e. failure to complete the search within the 100ms limit). According to these results, the optimal accuracy is achieved with a threshold of 0.4.

In the following section we are going to briefly report on the features that were applied to the prediction algorithm, and their impact on the prediction accuracy, taking into consideration the speed constraints.

Table 3.4.1: Taken from Koehn et al (2014, Table 1, p.3), Impact of threshold pruning to the accuracy and failure rate of the prediction algorithm to complete the search within 100ms.

Threshold	Accuracy	Fail
0.3	55.8%	4.5%
0.4	**56.1%**	**6.5%**
0.5	55.9%	9.0%
0.6	55.5%	11.06%
0.8	54.4%	17.1%
1.0	52.7%	21.7%

3.4.1 Refinements to Search graph based ITP (Koehn et al. 2014)

Starting off with pruned search graphs with a threshold of 0.4, the following 5 features were applied on the prediction algorithm:

3.4.1.1 Matching the last word of the user prefix

This is the lastMatched feature, but with an important addition. The last token of the prefix is searched not only in the matched prefix of the search graph, but also in the matched suffix, namely the prediction string. Furthermore, the last prefix token is searched for within a window around the last matched word, meaning that the last user prefix token could be up to a few words away from the last word of the matched prefix or matched suffix. Table 3.4.2 shows the results according to the number of words for the window, given a 100ms timeout.

Table 3.4.2 Taken from Koehn et al (2014, Table 2, p.3). Prediction accuracy within a 100ms timeout when searching for the last prefix word in a window around the last word that was matched in the matched search graph path.

Window	Accuracy
Baseline	56.1%
1 word	56.6%
2 words	56.9%
3 words	57.2%
5 words	57.8%
Anywhere	**59.1%**

From the results it is apparent that not restricting the search within a window, but allowing matching of the last prefix word anywhere on the path, boosts performance up to 59,5%.

3.4.1.2 Case insensitive method

The hypothesis behind case insensitive matching is that some mismatches between words matter less than others, and words with different cases could be treated as the same (e.g. President vs president).

At this point we should note that the input to the prediction algorithm is truecased[1] and tokenized. However, as Koehn (Koehn et al., 2014, Table 3, p.3) demonstrates, case-insensitive matching leads to a *decrease* in accuracy over the previous baseline (58.7% vs 59.1% for matching the last word). This is explained due to the high computation cost that is added to the algorithm.

Furthermore, exactly because the input is truecased, in some cases casing may actually be important for a correct prediction, as an uppercased word may be a proper name as opposed to a lower cased noun.

3.4.1.3 Approximate word matching

This feature aims in capturing matched words that differ by the user prefix by number, case, or even spelling inconsistencies. If a word typed by the user differs from the matched word by a few letters, it is considered a lesser error. The letter dissimilarity is computed from the ratio of letter edit distance to the length of the shorter word. For example, “cat” vs “cats” needs 1 insertion, and the length of “cat” is 3, so the dissimilarity score would be 1/3=0.33. Table 3.4.3 shows the accuracy of the algorithm when setting a maximum dissimilarity under which the mismatched words are considered as a lesser error (only half the edit cost of the other edit operations). With a threshold set to 10%, accuracy jumps to 60.6%.

[1] Truecasing is the conversion of the capitalization of words/sentences into their proper form. In Statistical MT, instead of lowercasing the input, the truecasing model is used to find the natural case of the word. The truecasing model is simply a list of words and the frequency of their different forms. For instance, in “the Golden Gate Bridge” “Golden” is capitalized because it is a proper name, whereas “golden” is more frequently found in its lowercase form.

Table 3.4.3 Taken from Koehn et al (2014, Table 4, p.4). Prediction accuracy within a 100ms timeout when setting a threshold for a maximum dissimilarity.

Maximum Dissimilarity	Accuracy
Baseline	59.1%
30%	60.2%
20%	60.4%
10%	**60.6%**

3.4.1.4 Stemmed matching

Stemmed matching also aims to match prefix words that differ only by a few letters from the word matched. The difference is that emphasis is given only on edit operations that belong to the ending (suffix) of the word, whereas the leading characters (the stem) stays intact. This feature is particularly useful for morphologically rich languages that use verb conjugation, noun declension etc. According to Koehn et al (Koehn et al, 2014, Table 5, p.4) this approach does increase accuracy over the baseline (59.3% over 59.1% after matching the last word, in Section 3.4.1.1), but it does not perform as good as the general approach in Section 3.4.1.3 (Approximate word matching).

3.4.1.5 Word completion

Autocompleting the word the translator types is an equal important task for an interactive translation tool as the word prediction. For instance, if the prediction algorithm comes up with the word “president” over “chairman” but the user types the letter “c”, then it should autocomplete with the latter.

Furthermore, if the last prefix word cannot be autocompleted with any word in the matched path, then entire vocabulary of the unpruned searchgraph is explored for potential matches (“desperate matching”). If more than one words match, the most probable one will be returned to the user as the autocompletion. According to Koehn et al (Koehn et al, 2014), this approach, combined with the approximate word matching and last word matching over the baseline, leads to 60.5% accuracy for a 100ms timeout.

3.4.2 Final conclusion

The 5 features in Section 3.4.1 were shown to improve the prediction accuracy, which indicates that giving emphasis on the last word of the user input can improve the results.

Furthermore, given that the majority of the failed predictions occur due to the short time span given, improving the algorithm itself can allow the exploration of more features.

Last but not least, it has to be kept in mind that interactive translation prediction depends on the search graph, which depends on the decoder. Therefore, the MT engine is also crucial for an accurate prediction, as it increases the search space.

Chapter 4.
Human Evaluation of CAT tools and UI

In the previous section we evaluated automatically and analysed the accuracy of the prediction tool. However, suggestions for improvement can only be given by the end users, which in this case are the translators themselves. Human evaluation of CAT tools and different modes of editing are an important complement to automatic evaluation measures such as the ones given in Section 3. Therefore, a practical evaluation, a.k.a. field trial study, needs to be conducted in order to evaluate the usability of the prediction tool and to measure the cognitive effort of the users, while comparing the two translation modes (simple Post Edition vs Interactive Translation Prediction). The analysis of the method and results is given in Section 5.

4.1 Translation Process studies

A number of academic studies have been run in the past to evaluate translation from scratch vs post editing of MT output, but rarely to evaluate different ways of post-editing as we do here.

Studies that evaluate translation from scratch vs post edition are very important, as they have shown that post-editing Machine Translation output can be more efficient than translating from scratch (Plitt and Masselot, 2010; Federico et al., 2012), both in terms of speed and quality. Details on several process studies are given on Section 4.1.2. As mentioned before, in post-edition the translator is presented with an initial Machine Translated hypothesis and he is requested to edit it if necessary (hence the name, "post edit") to reach the final result

One might think that MT lowers the translation quality due to the bias that can be caused when the translators are presented with an automatically translated output, especially when the automatic translation is too literal, such as when idioms or collocations are translated word for word. For example, the English idiom "It is the early bird that gets the worm" translates into Spanish as "A quien madruga, Dios le ayuda", the literal back translation of which is "God helps he who wakes up early". However, Machine Translation (e.g. Google Translate[1]) gives the literal translation of the English idiom: "Es el pájaro temprano que consigue el gusano", which rhymes well but is not a Spanish proverb. Fortunately, if the idiom has been seen in the training data of the phrase-based Machine Translation engine, then it can be translated successfully. A typical example used in MT is "It's raining cats and dogs", which is correctly translated into Spanish as "Lueve a cántaros" ("It is raining pitchers"). Interestingly, when giving "*It is* raining cats and dogs" (instead of "it's") as input to Google Translate, the output is the literal "Está lloviendo gatos y perros". A potential explanation would be that "It's raining cats and dogs" exists in the Phrase Tables as is, whereas "It is raining cats and dogs" does not, and

[1] translate.google.com, Accessed 19 July 2014

the individual translation options regarded by the decoder score the literal translation higher.

The examples above enhance the worry that machine translated output lacks fluency by being too literal. Furthemore, Machine Translation often lacks context, which means that reference resolution or consistency in translation are even harder problems for MT.

However, contrary to these popular assumptions that a less fluent output can harm the final translation, in general human translation quality does not decrease with the use of MT (Carl et al, 2011). On the contrary, in a process study for the evaluation of Caitra (Koehn, 2009b), it was found that the (non-professional) participants of this web based tool were generally not only more efficient in terms of time, but they also produced better translations in terms of i) fluency, which can be subjective, and ii) QA metrics, which are more objective, as they are a summary of misspellings, typographical, stylistic, grammatical and syntactical mistakes found in a document, that are scored according to whether they are minor, major or critical mistakes. Translation process studies use either automatic scores or human evaluations.

Following similar process studies, on professional translators this time, Plitt et al (Plitt et al, 2010) showed similarly encouraging results; professional translators were also faster and performed fewer errors. More details are included in Section 4.1.2.

4.1.1 Usability Tools for Translation Process Studies

A usability tool that is popular among UX studies is the Think Aloud Protocol (TAP), first introduced by Clayton Lewis (Lewis, 1982). Nielsen (Nielsen, 1993) was also a great supporter of this usability tool and included it in his book "Usability Engineering" that we mentioned in previous Sections. As the name implies, in usability tests where TAP is applied, users need to think out loud and describe actions, thoughts or even concerns while using the tool in test. The benefit of this particular method is that it is very cheap, flexible, and easy to learn.

Jääskeläinen (Jääskeläinen, 2001) introduced Think Aloud Protocols to translation process studies in 2001. Similarly to any application of TAP, in these studies the translator is asked to verbalize the thoughts that led to typing a specific translation or to pause for a long period. However, translation is a cognitively demanding process, and being asked to explain all actions while translating interferes with the translation process. Furthermore, TAP requires specific types of testers, who do not mind monologuing; all in all, the TAP setting is rather unnatural, even though easy to learn, because people are not used to talking to themselves while in the middle of a process. For this reason, Jakobsen (Jakobsen, 2003) supported a few years later that this way of performing process studies slows down the translation speed and interferes a lot with the translation process. Therefore, it was quickly replaced by more indirect ways of measuring user activity, such as the User Activity Data (UAD).

In the last two decades, the increase of usage of computers by human translators allowed new insights of the translation process, and process studies based on User Activity Data (UAD) emerged (e.g. Fraser, 1996). Researchers started using tools such as Translog (Jakobsen & Schou, 1999) that allow logging of keystrokes and the total time

needed for a translation of a sentence. Analyzing these logs can help understand the human cognition, namely the way translators think, and which parts are easy or difficult to translate. Furthermore, UAD has the advantage of being objective and reproducible.

Apart from the logging of keystrokes and mouse movements (Langlais et al, 2010), several studies (e.g. Sharmin et al., 2008) started using eye-tracking data to determine where the translators spend most of their times on. For instance, it was found that translators spend more time looking on the target text than the source text (Sharmin et al., 2008).

Regarding the usability of the platform that is used for translation, it has to be taken into consideration that the quality of the MT output is highly correlated to the perception of the usability of the workbench as Kumaran et al. (2008) noted. More specifically, they found that, for the wikiBABEL system they implemented, users provided positive feedback about the system only when the MT output they were given to post edit had high quality, i.e. a BLEU score of about 30. On the contrary, when the MT quality was low (around 15 BLEU), the professional translators tested in the User Study were deleting the whole sentence and were rewriting it from scratch, whereas amateur translators kept using the words that were in the MT output, but they spent a lot of time rearranging the order so as to form the final translation. An explanation regarding the different approach of amateurs vs professional translators is that amateurs are that a) amateur translators are in more need of translation aids, but at the same time b) they are more open to new tools, as it doesn't contradict to what they have learned to use so far. According to Kumaran et al, amateur users were in overall more in favor the system than the professional translators, but in both cases it was concluded that providing low-quality MT is worse than providing no translation at all, as translating from scratch slows down the translation speed and leads to user frustration. This may seem as contradictory to our claim that MT helps translators produce more throughput, but low-quality MT could be hidden on the sentence, not document level.

For that reason, automatically generated Confidence Estimation scores could be used as indicators to allow the translator to know whether a certain segment of the MT output is good quality or not, so as to place more emphasis on it or choose to ignore the MT suggestion completely. Alternatively, this information could be used in the back end of the CAT tool as well, as a mean to prevent displaying an automatic translation (on the sentence level, as we said above) to the user if the quality is too low. However, to our knowledge, MateCat[1] is the only CAT tool that has integrated confidence estimation features when displaying the MT alternatives.

There have also been some attempts to indicate the segment quality in the CasMaCat project (Figure 4.1), but the participants of the user study (Carl et al, 2013) reported that, even though the idea was very interesting, they quickly lost confidence on it after the first false positive segment was marked as red. For example, Named Entities and Acronyms

[1] www.matecat.com, accessed 19 July 2014

were marked as red by the MT and they were not filtered out by an external tool on the UI. Therefore, users quickly learned not to trust the output.

Figure 4.1: User interface of the CasMaCat workbench using Quality Estimation scores. Words highlighted in red have a large probability of being incorrect translations, whereas words highlighted in orange are dubious, but could still be correct.

Interestingly, this might also be a UX problem. Had the information been displayed more discreetly on the UI (e.g. by underlining potential errors, as it is done on most text editors), the users might have been less judgmental regarding false positives.

4.1.2 Conclusions from previous Translation Process Studies

Translation Process studies are generally influenced by many factors, which makes direct comparisons of i) evaluations of tools and ii) cognitive effort assumptions difficult. Some of the above-mentioned factors are:

- The MT Quality, especially when comparing older (e.g. Krings, 2001) and more recent (Plitt et al., 2010) studies
- The language pair(s) chosen. Translation between very different languages is harder because they may differ greatly in word order. Especially when the target language is morphologically rich (e.g. Hungarian or Finnish) and the source language is not (e.g. English) translation becomes more challenging, which has impact on the MT quality, as expressed by BLEU score. In the EuroMatrix Online Evaluation page[1] the results show that all BLEU scores for Finnish as an output language are significantly lower than when translating into any Indo-European language. For example, English to Spanish has a BLEU score of 30.16 in the evaluation, whereas English to Spanish only 13.00.
- The translator's level of expertise: amateur bilingual (as in Koehn, 2009b or in our user study in Section 5), student of translation studies, or professional translator (e.g. Plitt et al., 2010))

[1] http://www.statmt.org/matrix/, accessed 20 October 2014

- The tool itself and the prior familiarity of the users with it. For example several translators have worked with various post editing tools, but others may have only been translating from scratch.
- The duration of the user study. Longitudinal studies handle better the familiarity and learning curve factor, but due to their high cost they do not occur as often.

TransType (Foster et al., 1997) was the first interactive system that was based on SMT. The UI (Foster et al, 2002) offers predictions in the form of an autocompletion drop down menu (Figure 4.2). In the TransType UI, the source text is displayed fully in the upper half of the screen, whereas the text editor is displayed in the bottom half. As the user types his translation, a drop down text with multiple auto completion suggestions shows up, and the translator can either accept one of those or continue typing. According to (Foster et al, 1997), TransType is capable of correctly predicting over 70% of the typed characters, this does not mean that it saves 70% of translator's time.

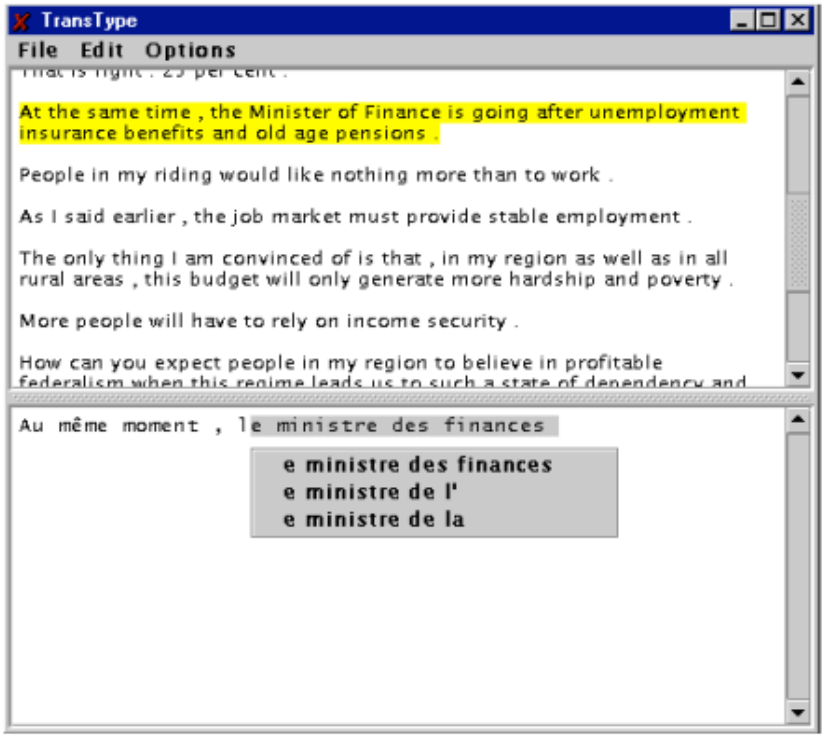


Figure 4.2: Taken from Langlais et. al (2002): an example of a user session using TransType. An animated screen dump of a short translation session is given at: http://www-rali.iro.umontreal.ca/ttype-proto.en.html

In 2000, during the first evaluation of the TransType project (Langlais et al, 2002), ten French native translators were asked to work for an hour on a realistic working environment using the TransType tool, where they could obtain automatic completions for their translations.

Two of the volunteer participants were translation professors, two were translation students and the others were professional translators (Langlais et al, 2002). The authors claimed that the users were anxious about being evaluated for their performance, and therefore there was no automatic or human evaluation done on the translation, only on their speed. However, a brief look at the translations produced did not show any issue with quality, according to Langlais.

Regarding results, the users themselves reported an improvement in their performance and the quality study revealed that 90% of the participants liked the tool and would be eager to use it more often. Nevertheless, the logs showed that raw productivity decreased by 17% compared to translation from scratch. In fact, only one translator got faster while using the tool. However, this could be explained by the fact that the users often chose to type the translation, even in the cases that the suggestion was correct. Another factor could be that the multiple suggestions take long to read, and, especially when they are incorrect, they increase the translator's cognitive load. But most probably, the decrease in speed can be attributed to the learning curve and the user's need to get used to the new way of translating; even the translators themselves were confident that after proper training with the tool, they would be able to increase their productivity.

In 2004, TransType2 (TT2) (Esteban et al, 2004) was proposed, as a continuation to the TransType project. The UI is quite similar to the TransType project (Figure 4.3), but the source text is on the left whereas the target text (editing box) on the right, which allows for clearer source-target sentence alignment. The goal of TT2 was to improve translators' productivity by decreasing the number of keystrokes needed to type a translation.

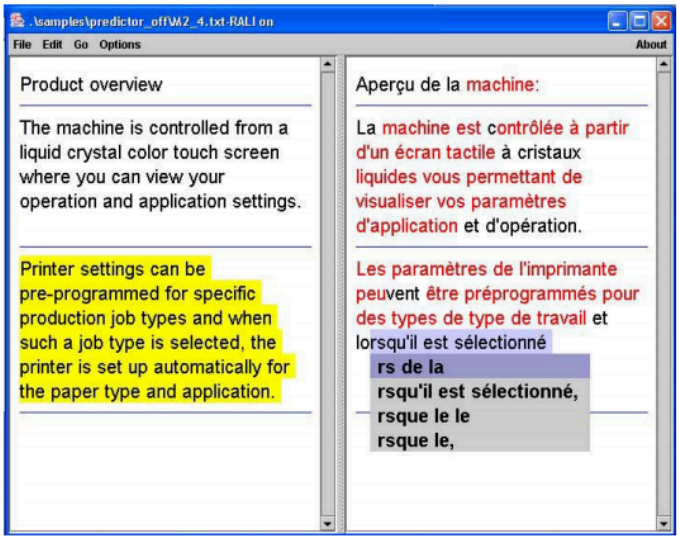


Figure 4.3: TT2 interface, taken from (Esteban et al, 2004, p.2). As the user types, the dropdown box indicates the suggestions, which can be accepted either by clicking or using the keyboard. In this picture, words that were accepted by the translator are marked in red

Within the TransType2 (TT2) project, Barrachina et al (2009) extended the interactive system to provide real interactive post-editing, in the sense that the users were provided with a partial initial MT suggestion, that the user could edit and accept. Whenever the users edited the suggestions even partially, the system recomputed its suggestions in the way we described in Section 2. The authors ran a simulated translation process study, similar to the automatic evaluation in Section 3, for 6 language pairs, English<->Spanish, English<->French and English<->German. The tasks were Xerox (Xerox printer manuals) and EU (Bulletin of the European Union). By simulated we mean that they did not have users evaluating the system through the UI, but instead they used translated references to assess the tool. The goal of this is to estimate the effort needed by a real user, but this does not take into consideration the UX factor, or the cognitive load caused by the display of multiple suggestions that need to be read before they are accepted or rejected. In contrast to (Langlais et al, 2002), Barrachina et al. found a reduction of up to 80% in the typing effort of interactive post-edition compared to translating from scratch. However, a real user evaluation should be used to verify the results.

At the same year, Koehn (Koehn, 2009b) evaluated Caitra, the CAT tool he developed, using ten non-professional translators that were either native in French or in English. The tool provided 3 types of assistance (Figure 4.4): auto completion of translations, alternative translation options for each source segment and phrase, and simple post editing of output. Caitra is web based, and it logs every keystroke and mouse click of the user for further analysis.

Figure 4.4: Caitra's UI. On the top left screen, the user can see the source text with the various numbered segments. The editor can only edit one segment at a time. Note

that this particular screen shot is not taken from the field trial, as in this case the source language is German and the target language English.

The participants were asked to translate 192 sentences each, namely news stories from Le Devoir, Le Figaro, Les Echos and Liberation from French into English, while their interaction with the tool was examined. Machine Translation from French to English has high quality, similar to the English Spanish language pair that we are using here. This study is also the first direct comparison of the two types of assistance, post-editing and Interactive Machine Translation. Given that Caitra is web based, the participants were allowed to complete the task at their own convenience, within a two-week period.

Participants completed their tasks under 5 types of assistance: i) unassisted, ii) post-editing MT, iii) using translation options, iv) prediction (autocompletion) v) a combination of options and predictions ("prediction plus options").

In order to evaluate human quality, the author recruited 6 human reviewers. All 10 translations for each sentence were displayed on the same screen. Notably, the human reviewers gave surprisingly low levels of correctness (50% on average). When analysing the reviews manually, Koehn (2009b) was left with the impression that the reviewers were too critical, and, due to the fact that they were presented with 10 translations at a time, they might have been tempted to rate half as good and half as bad. Therefore, when it comes to human evaluation, how feedback is requested is also of major importance.

Another interesting remark is that for many sentences, each translator came up with a different translation, which supports the suspicion that human translation has very high variability, and is therefore hard to evaluate objectively.

The participants were also asked to give feedback in the form of a multiple-choice questionnaire. In the question "Which of the five conditions did you enjoy the most?", *unassisted* and *post-edited* were chosen only once, *options* twice, *prediction* twice and *prediction plus options* three times. Similarly, when asked in which condition they felt most accurate, *post-edited* was chosen once, *prediction* once, *options* twice, and *prediction plus options* five times. In general, the participants enjoyed the other types of assistance much more than simple post-editing.

The results indicate that, in general, the translators were not only faster, but also better when translating with assistance as opposed to translating from scratch. Some translators even cut their translation time by more than half. However, different translators have very different backgrounds in terms of familiarity with translation or CAT tools.

According to the results, they can be grouped into three categories: slow translators, fast translators and refuseniks. In the slow translators group there were 4 participants that required more than 5s per input word when translating without assistance. Half of them produced low quality texts when unassisted (35% and 16% correct according to the human evaluation) and became much faster and better with assistance (41-61% and 18-34% depending on the assistance). The other two slow translators produced texts of average quality, and became faster with assistance, but did not increase their performance. In the fast translator group there were only 2 participants. Both use the assistance offered and become even faster, however one of the fast translators produces

really low quality (23%) and becomes better with assistance, but still below average (30-45%). Last but not least, in the refuseniks group there are the last 4 participants who rarely use the assistance provided. However, 2 of the highest ranked translators are in this group, and they become even better (from 68% unassisted to as much as 79%) when post-editing.

The author mentions as well the learning effect that is observed due to the unfamiliarity with the types of assistance provided, given that all translators are amateurs and that there are new features tested (options and prediction). Koehn (Koehn, 2009b) states that, even though the assistances offered were intuitive, translators may need to get used to it if before they become fully proficient. Indeed, Figure 4.5 shows that translators speed up as they translate more sentences and they become more familiar with the task and the translation tool. He summarizes that each translator submitted 32-46 sentences per hour with each type of assistance, and that the speedup is more apparent on post-editing. On the contrary, and as expected, there is no gain in speed after the initial start-up bump when translating without assistance.

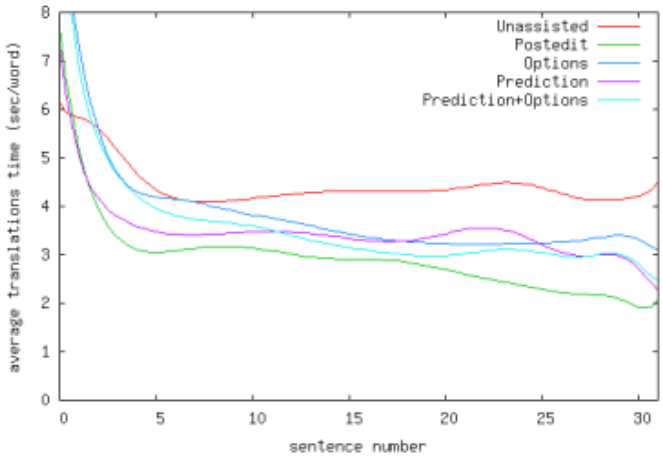


Figure 4.5: Taken from Caitra's process study (Koehn, 2009b p. 23, Figure 9). The graph shows the reduction in translation time per assistance type that comes with experience, after the completion of a certain amount of sentences

Finally, regarding interactive translation vs post-editing, results do not indicate that interactive translation increased either the translator's speed or quality. On average, they were faster by 39% when post-editing compared to translating from scratch, whereas they were 16% faster when given translation options, 27% faster when given predictions, and 25% faster when using both options and predictions. This suggests that translators are fastest when post-editing, and obtain highest translation performance when using all types of assistance (post-editing, plus options, plus prediction). It is also surprising that participants ranked post-edition as less useful in the feedback, even though it lead to an average boost of speed, and is therefore subjectively useful.

Plitt and Masselot (Plitt and Masselot, 2010) conducted a two day field trial with 12 participants who were translating from English to French, Italian, German, and Spanish. The authors selected a subset of real data collected in 2009 from their software company, Autodesk, which localizes its products from English into various languages. The MT engine was a Moses system (Koehn et al., 2007) trained only on the full Autodesk localization data. The authors created their own workbench, which, as they stated, was highly inspired by Caitra (Koehn, 2009). The workbench (Figure 4.6) was displaying the source, as well as target phrases in PE mode, and it recorded the total edit time, the number of edit sessions and the number of key strokes for each sentence.

The 12 participants (3 for each language pair) were not provided any training, but they were given simple post-editing instructions. All participants started translating from scratch (FS) and they moved to post-edition (PE) in the second phase. Nevertheless, to remove the sentence factor, the texts were mixed so as to make sure that all sentences were translated both from scratch and using post-edition (by different translators).

The quality evaluation chosen was the the usual QA analysis which is run in translation agencies, where reviewers are requested to report minor, major and critical errors in the final translated output. The QA team were not aware of the mode used to translate (FS vs PE) so as to eliminate any bias. The total source tokens processed in this study were 144,648. The authors found high variance across translators, but they concluded that

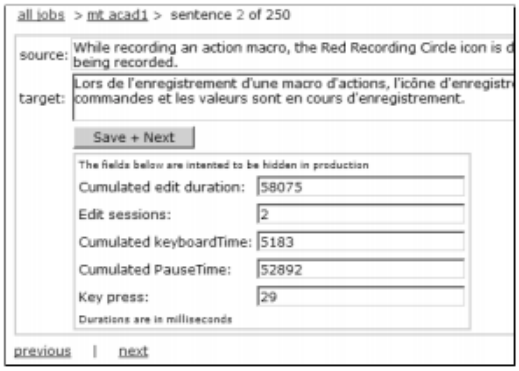


Figure 4.6: Screenshot of the Plitt and Masselot workbench (Plitt and Masselot, 2010, p.2, Figure 1). Time recording fields were hidden from translators.

Machine Translation resulted in speed increase for all of them, ranging from 20% to 131%; by 100% productivity gain, the authors indicate that their throughput has doubled. On average, MT allowed translators to improve their throughput by 74%. Therefore, Plitt and Masselot concluded that Post Edition saved 43% of the translation time.[1]

One recent study (Sanchis-Trilles et al, 2014), which is longitudinal as opposed to most of the translation process studies introduced before, is the evaluation of the CasMaCat (Alabau et al., 2013) tool, which is our CAT tool of interest as we develop and evaluate the

[1] 1 - 1 / 1 + 0.74 = 0.43

prediction tool around it. CasMaCat shares much functionality with its precedent, Caitra (Koehn, 2009b), but it offers word alignments for the highlighting of source and target words, a biconcordancer for word lookup, a paraphraser for alternative phrase suggestion, and, mainly, a cleaner UI. Furthermore, it supports adaptive learning so as to learn from user feedback and stop repeating mistakes over and over again. The main goal of CasMaCat is to study the cognitive processes of translation using UAD and eye-tracking information.

Regarding Interactive Translation Prediction, CasMaCat offers several ways of presenting the suggestions: i) via the floating prediction that displays the next 3 tokens, which is used in this study (Section 5), ii) by displaying the whole prediction path in a light grey color, whereas the translation that the user has typed or accepted is displayed in black (Figure 4.8), iii) same as the second option but the suggestions are displayed in black as well (Figure 4.7). In i) and ii) the black text belongs to the human editor, and it is never modified by the CAT tool, whereas the light grey area is the suggestion, and it can change interactively; plus, once a grey text is accepted, it becomes black.

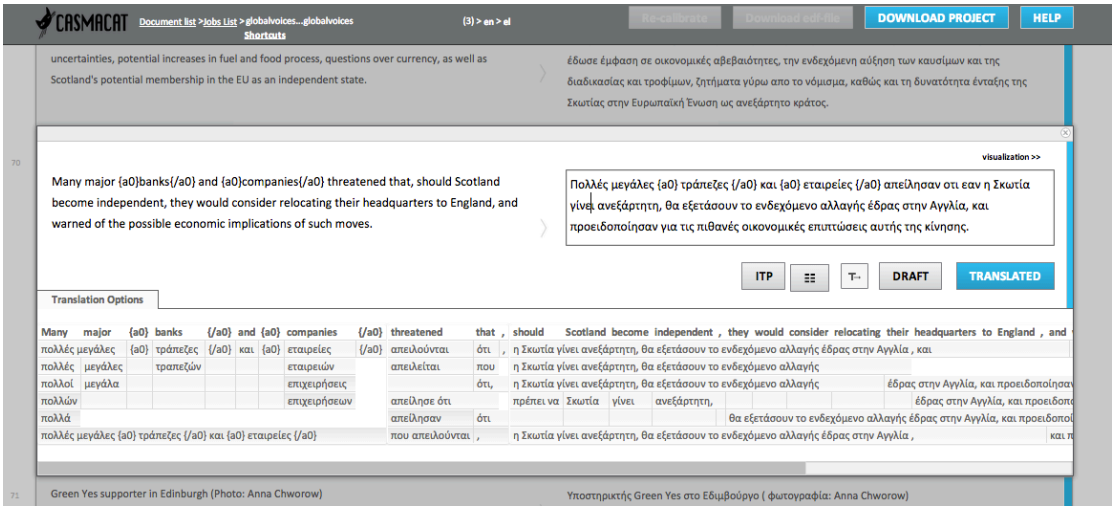


Figure 4.7: The CasMaCat UI with translation options and ITP enabled. Note that the screenshot is not taken from the User Study, as here the target language is Greek.

The goal of the longitudinal study was to compare ITP, as described in Section 3, vs conventional post-editing, both in terms of speed and quality.

Nine native Spanish professional translators were asked to translate news commentary corpus from WMT12 (Callison-Burch et al, 2012) from English to Spanish, and four reviewers evaluated their translations.

Figure 4.8: AITP with limited prediction horizon

The news commentary texts consisted of 30-63 segments each, summing to an average of 1000 words per segments, and the total number of segments was 9, divided in three different datasets. Each dataset needed approximately 3.5 hours to be post-edited.

In this translation process study, 3 conditions were tested: conventional post-editing (PE), Interactive Translation Prediction (ITP) and Advanced ITP (AITP) which included a few more visualization options:

a) visualization of MT confidence estimation (as explained in Figure 4.1),

b) limited prediction length; predictions are displayed only up to the first word of low confidence (Figure 4.8) , in order to decrease the cognitive load of users who are otherwise requested to read long predictions.

c) word alignment information between the source and the target.

d) visualization of user edits, to help the translators locate the changes that were introduced by him vs. the system's suggestions.

In this process study, participants were split into conditions and datasets so that each text was translated by 3 different translators in each of the three conditions. The exact task assignments for all nine editors can be seen in Figure 4.9.

	Text								
	Dataset 1			Dataset 2			Dataset 3		
	T1.1	T1.2	T1.3	T2.1	T2.2	T2.3	T3.1	T3.2	T3.3
Segments	49	30	45	63	55	51	59	61	47
Source words	952	861	1121	1182	1216	1056	1396	1427	1258
Editor 1	PE	ITP	AITP	ITP	AITP	PE	AITP	PE	ITP
Editor 2	AITP	PE	ITP	PE	ITP	AITP	ITP	AITP	AITP
Editor 3	ITP	AITP	PE	AITP	PE	ITP	PE	ITP	PE
Editor 4	ITP	AITP	PE	AITP	PE	ITP	PE	ITP	AITP
Editor 5	PE	ITP	AITP	ITP	AITP	PE	AITP	PE	ITP
Editor 6	AITP	PE	ITP	PE	ITP	AITP	ITP	AITP	PE
Editor 7	AITP	PE	ITP	PE	ITP	AITP	ITP	AITP	PE
Editor 8	ITP	AITP	PE	AITP	PE	ITP	PE	ITP	AITP
Editor 9	PE	ITP	AITP	ITP	AITP	PE	AITP	PE	ITP

Figure 4.9: Taken from (Sanchis-Trilles et al, 2014, p.7, Table 2). The table provides full information regarding the study setup (number of segments, words, type of assisting mode per editor and text)

It is important to note at this point that CasMaCat is a web based tool that can log user activity (UAD) in detail, and with exact timing information: key strokes, mouse activity, and even the user's gaze if it is used in combination with an eye tracker. Before starting the translation tasks, participants were given some time to familiarise themselves with the tool under all three translation modes, and to check all visualization options, so as to decide which ones they would enable in AITP mode. Furthermore, given that CasMaCat is a web-based tool, participants were allowed to complete Datasets 2 and 3 at home and deliver them over the Internet, but they were requested to complete the first dataset at

the office of Celer Soluciones SL in Madrid, where eye tracking devices were available. Gaze and logging activity of the reviewers was also logged. After each session, participants were asked to complete a questionnaire that measured their satisfaction.

The authors analysed the inter-keystroke pause duration from the translation logs to elicit information regarding the cognitive effort of the translators. As the authors cite, based on cognitive language processing and production theory (Alves and Vale, 2009; Carl 2012), pauses between 0 and 5 seconds are used to segment the translation process into "typing" and "processing" units. In order to analyse the various pause durations, the authors generated a Pareto chart (Figure 4.10) to display the relative contribution of groups of interval durations (0-10s, 10-20s, .., 220-230s) between keystroke activities. This way of display was specifically chosen because Pareto charts are used to extract the most important factor among a large set.

As it can be seen in Figure 4.10, pauses between 0-10s account for for more than 50% (in fact 58%) of the total processing time. This also means that filtering only for pauses up to 10s would leave 42% of pauses unaccounted for. Therefore, according to this analysis, Sanchis-Trilles et al (2014) set the threshold to 200s, to be able to take into consideration 95% of the total intervals.

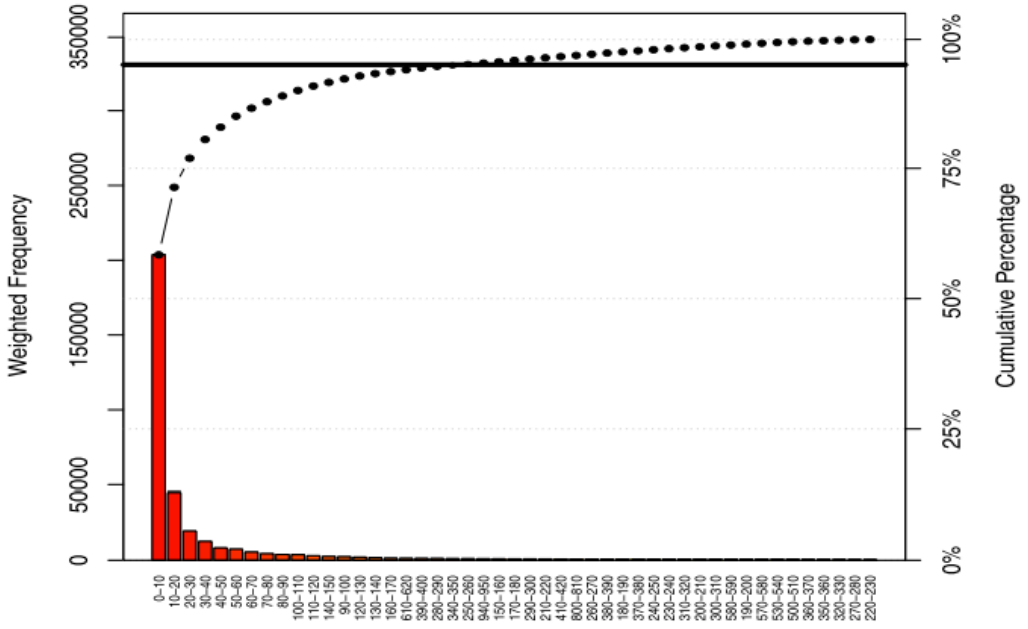


Figure 4.10: Taken from (Sanchis-Trilles et al, 2014, p.11, Figure 5). This Pareto chart shows the weighted frequency of pause durations between typing activities, within translation sessions

Furthermore, according to this analysis, two kinds of filtering were used to process the activity logs (Sanchis-Trilles et al, 2014):

- **Kdur**, which is the total durations of continuous typing activity (events that are at most 5s apart)

- **Fdur**, which is the total processing duration, ignoring pauses longer than 200s

Figure 4.11 shows the average segment processing times in seconds for the three processing modes: PE, ITP, and AITP. The processing times are Kdur and Fdur described above, plus Tdur, which is the total processing duration.

System	Tdur	Kdur	Fdur
PE	104.0	21.7	73.0
ITP	80.7	27.0	77.0
AITP	117.1	29.6	92.4

Figure 4.11: Taken from (Sanchis-Trilles et al, 2014, p.12, Table 4). Shows the average processing time per edit mode (PE, ITP, AITP) for all datasets

According to the filtered measures (Kdur and Fdur), PE leads to slightly shorter processing times. However, the difference between PE and ITP processing times are not that big: for Fdur, ITP is 5% slower. According to the authors, one possible explanation for the slightly bigger difference in terms of Kdur is that in interactive translation, users are required more often to perform short post-edit operations. Tdur values are high due to a small number of extreme outliers, and they are given here as a reference.

In order to take the learning curve into consideration, the authors analyzed the times of the first Dataset, that was completed at the office, versus Dataset 2 and 3 that were completed from home, after the translators had gained a certain familiarity with the tool. Indeed, Figure 4.12 shows a reduction in processing time for both Kdur and Fdur. It is important to note that the speed-up is higher for ITP and AITP when compared to simple post-editing mode. As the authors concluded, this correlates well with the assumption that the translators were initially more efficient with PE due to familiarity.

System	Kdur		Fdur	
	Office	Home	Office	Home
PE	27.7	19.6	88.0	67.3
ITP	35.1	24.8	94.7	71.9
AITP	37.5	27.2	111.9	87.8

Figure 4.12: Taken from (Sanchis-Trilles et al, 2014, p.12, Table 5). Shows the average processing time per edit mode (PE, ITP, AITP) when working at the Office (Dataset 1) and from home (Datasets 2 and 3)

Furthermore, the authors analysed the number of text insertion and deletion by the user. The hypothesis is that interactive translation will require less manual insertions. Indeed, as shown in Figure 4.13, ITP required fewer manual actions.

System	Office	Home	All
PE	114.9	134.6	131.3
ITP	109.6	127.2	123.6
AITP	143.2	137.0	132.6

Figure 4.13: Taken from (Sanchis-Trilles et al, 2014, p.12, Table 6). Shows number of insertions and deletions by the users for each system and when working from the office, at home, or for all the sessions.

In order to assess quality, the authors analyzed the edit distance between the reviewer's corrections and the original documents, meaning that they took the number of insertions, deletions and substitutions and normalized by the number of all words (so including the correct words). The analysis showed that all 3 systems resulted in translations of similar quality (90.7% for PE, 89.3% for ITP and 89.9% for AITP).

Last but not least, user satisfaction was elicited in the form of questionnaires. After each session, the editors were asked to answer the following questions:

- How satisfied are you with the translations you have produced? (Satisfaction)
- How would you rate the workbench you have just used in terms of usefulness/aids to perform a post-editing task? (Tool)
- Would you have preferred to work on your translation from scratch? (From scratch)
- Would you have preferred to work on the MT output without the interactivity provided by the system? (No ITP)

	Satisfaction			Tool			From scratch			No ITP	
	PE	ITP	AITP	PE	ITP	AITP	PE	ITP	AITP	ITP	AITP
Editor 1	3	4	4	3	4	4	No	No	No	Yes	No
Editor 2	4	4	4	3	2	4	Yes	Yes	Yes	No	No
Editor 3	3	3	4	3	3	4	Yes	No	No	Yes	No
Editor 4	4	4	5	3	4	4	No	No	No	No	No
Editor 5	4	3	4	4	4	3	No	No	No	Yes	No
Editor 6	5	5	5	3	3	2	No	No	No	Yes	Yes
Editor 7	3	4	3	2	1	2	Yes	Yes	Yes	Yes	No
Editor 8	4	4	3	2	2	3	Yes	No	No	Yes	Yes
Editor 9	4	4	4	1	4	3	Yes	Yes	Yes	Yes	No

Figure 4.14: Taken from (Sanchis-Trilles et al, 2014, p.16, Table 11) User satisfaction ratings, with 1 being the lowest rating and 5 the highest.

Figure 4.14 summarises the results that show different levels of satisfaction for different systems. Participants 1, 3 and 4 seem to enjoy more the translating iteractively than post-editing. Generally, regarding the tool, interactive translation modes are rated

higher than PE, even though, as the authors point out, 7 out of 9 translators stated that they would have preferred not working with the interactive tool in the ITP system. This changes for the Advanced ITP system, where only 2 translators kept thinking that they would have preferred to work without interactive features.

To summarise, in this longitudinal study Sanchis-Trilles et al (2014) show that even with little training interactive translation can be as fast as conventional post-editing. Moreover, ITP requires less keystrokes to reach the final translation and ITP tools lead to high user satisfaction.

In the following Section, we are going to repeat the user study, but with non-professional translators and a different visualization option for the Interactive Translation Prediction mode. The goal is to test the user satisfaction and the usability of this alternative ITP mode.

Chapter 5.
User study

As mentioned in Section 4, suggestions for improvement of a system can only be given by the end users, which in this case are the translators. In order to evaluate the usability of the prediction tool and to measure the cognitive effort of the users, a practical evaluation (field trial) was conducted using 6 non-professional translators, namely native Spanish speakers that are fluent in English.

The goal of this process study is to get feedback on the usability of the interactive tool, and more specifically the "floating prediction" display of suggestions that was not tested in the longitudinal study (Sanchis-Trilles et al, 2014). Therefore, in this study the goal is similar; to measure the usefulness of Interactive Translation Prediction (ITP) over the classic post-editing (PE) approach.

In the case of ITP, due to the constantly changing suggestions produced by the tool, there is a general belief (e.g. Alabau et al., 2012) that such an interactive way of translating increases the cognitive effort of the translators, and therefore does not increase the translator's productivity as much as conventional post editing. However, the hypothesis is that the interactive prediction is suggesting translations closer to the translator's needs, and can therefore assist by resulting in less typing effort. Furthermore, the predictions can be used as implicit check against errors.

With this study we aim to investigate i) whether ITP is faster than PE, and ii) whether ITP leads to higher quality translations.

5.1 Method

In this study, 6 native speakers of Spanish (all male, 25-31 year old) were asked to translate 36 isolated sentences from English to Spanish using two different conditions in the CasMaCat workbench. In half of the cases, they were simply asked to post-edit (PE) the MT output (Figure 5.1), and in the rest to translate the text using Interactive Post Editing, aka Interactive Translation/Text Prediction (ITP); namely to post edit interactively with the help of the prediction tool (Figure 5.2).

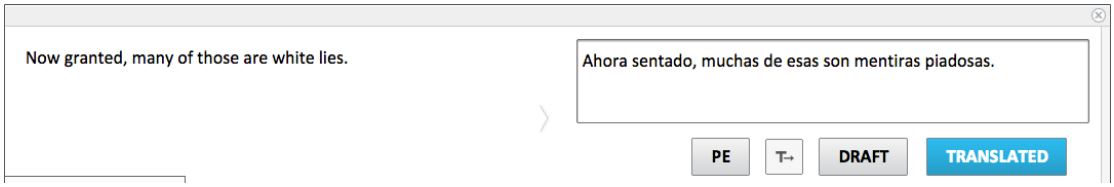


Figure 5.1 PE condition. The users are presented with the full best automatic translation that they need to post-edit

In the case of ITP, the participants were asked to use the interactive suggestions as much as possible and to accept them using TAB instead of keep typing. The reason for that was that the users might have found it easier at first to keep typing as they are used to, but in fact accepting a word instead of typing it would save them time. The assumption

was that they would easily get used to it, and this instruction would help reduce the effects that arise from the learning curve.

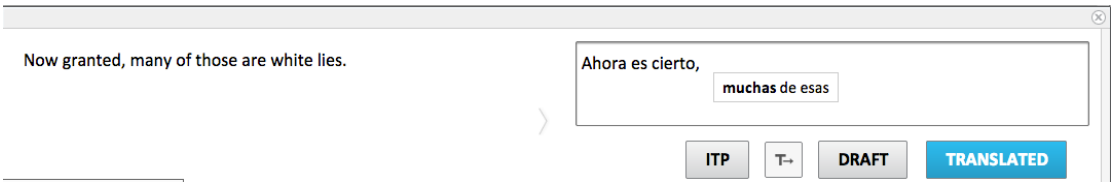


Figure 5.2 ITP condition. As the user keeps typing, the prediction shows up in the form of a floating suggestion. The user can accept the word in bold ("muchas") by using the TAB key.

Because the participants were not professional translators, they were allowed to look up a dictionary when the interactive suggestions or first-best machine translation were not enough to disambiguate the meaning. However, they were instructed to only use http://www.wordreference.com/es/ and they were discouraged from using any other external help in order to make sure they all had the same aid.

Using the CasMaCat logging feature, the user's typing speed was closely examined, along with the total completion time, the pauses and the keystrokes (UAD) they had to press in order to produce the final translation of a source segment. Furthermore, after the completion of the sessions, the participants were asked to give feedback on the usefulness of these two methods by completing a short survey (Section 5.2.3). The task was web based, so the participants could complete it in their own time and convenience, and the total task time was estimated to take about 40 minutes.

The language pair chosen for this task is English (source) to Spanish (target). The reason for this is that this language pair has a relatively high BLEU[1] score, as describe in previous Sections. For example, the winning system (ONLINE-B) of WMT12 (Callison-Burch et al, 2012) had a BLEU of 36 for the English-Spanish news task vs. 18 BLEU for the English-German news task. As mentioned before, a low MT quality could result in the users perceiving the translation tool as less usable (Kumaran et al., 2008), so we wanted to make sure that a low BLEU score is not a factor when comparing the two ways of editing the target text.

The dataset used for the source segments is a subset of FTD_2012_CasMaCat (more details on Section 3) that had already been post-edited in the first CasMaCat field trial study by professional translators. More specifically, the two news articles selected were taken from CNN and The economist.

[1] BLEU Score (Papineni K. et al, 2002) is an automatic process of evaluation of MT output. Ideally, human evaluation could be used for this task, but that would be time consuming and expensive. BLEU scores range from 0 to 1 (or 0-100%), and there is a correlation to human evaluation.

In each of the two conditions, the same set of 36 different texts, divided into two sets of 18 segments each, was translated six times by six different participants. As a result, each translator translated each text exactly once, and all users translated using both conditions (Table 5.1). In both cases, the translators were provided with an initial MT output, that they had to post process.

Table 5.1 Task assignments

	Dataset 1	**Dataset 2**
Segments	18	18
Source words (**tokens**)	289	352
Editor 1	ITP	PE
Editor 2	ITP	PE
Editor 3	PE	ITP
Editor 4	PE	ITP
Editor 5	ITP	PE
Editor 6	PE	ITP

5.2 Results

Following closely Sanchis-Trilles et al (2014), we started with the analysis of the user logs. Table 5.2 shows the information within a translation session that is stored by CasMaCat.

In theory, the total processing time of a segment is the time lapsed between the moment the post-editor opens the edit box of a specific segment, and the time he submits the translation as completed and exits the edit area. However, because the participants of this user study were asked to conduct the translation from home, in their own convenience and without time pressure, the logs show that some segments have very long overall processing time, with long pauses up to several hours, which suggests that the participants interrupted some sessions and returned to them later, or that they clicked on segments even before focusing on them to translate. For that reason, following the approach of Sanchis-Trilles et al (2014), the focus is done not on the total processing duration **(TDur)**, which is the duration of unit production time that are described on Table 5.2, but mainly on **Fdur**, the duration filtering for keystroke pauses larger than 200seconds, and **Kdur**, the duration of continuous keyboard activity excluding pauses larger than 5 seconds.

On table 5.3 we present the average processing time in seconds per document in terms of Kdur and Fdur using the two translation modes (PE and ITP).

Table 5.2 Translation/post-editing process data logged and stored by CasMaCat (Carl and Kay (2011); Carl (2012b; 2014)). CasMaCat also logs fixation units when an eye-tracker is available.

Keystrokes (KD): basic text modification operations (insertions or deletions), together with time of stroke, and the word in the final text to which the keystroke contributes.

Production units (PU): coherent sequence of typing, defined by starting time, end time and duration, percentage of parallel reading activity during unit production, duration of production pause before typing onset, as well as number of insertions and deletions.

Activity Units (CU): exhaustive segmentation of the session recordings into activities of typing, reading of the source or reading of the target text.

Source tokens (ST): as produced by a tokenizer, together with TT correspondence, number, and time of keystrokes (insertions and deletions) to produce the translation and micro unit information (see below).

Target tokens (TT): as produced by a tokenizer, together with ST correspondence, number, and time of keystrokes (insertions and deletions) to produce the token, micro unit information, amount of parallel reading activity during.

Alignment units (AU): transitive closure of ST-TT token correspondences, together with the number of keystrokes (insertions and deletions) needed to produce the translation, micro unit information, amount of parallel reading activity during AU production, etc.

Segments (SG): aligned sequences of source and target text segments, including duration of segment production, number of insertions and deletions, number and duration of fixations, etc.

Session (SS): global properties of the session, such as source and target languages, total duration of the session, beginning and end of drafting, etc.

Table 5.3: Average processing time (in seconds) per document and for all segments in terms of Kdur and Fdur when using the two different translating modes.

	Kdur		**Fdur**	
System	**Document 1**	**Document 2**	**Document 1**	**Document 2**
PE	29.15	15.01	168.24	97.78
ITP	32.43	39.48	136.40	121.59

Processing times, as shown in Table 5.3, are larger than the ones presented in Sanchis-Trilles et al (2014), which can be explained by the fact that the translators in our user

study are not professionals, nor familiar with any computer aided translation tool, and require more time to disambiguate the source text and edit the target.

Similar to the longitudinal study though, in both datasets post-editing the texts using PE is faster in the case of Kdur, because of the small post-editing steps that are required in the ITP mode, as mentioned in Section 4.

Regarding the total duration, filtering for pauses longer than 200 seconds (Fdur) gives similar results for both editing modes, with ITP being only slightly faster. The Fdur for both datasets in post-editing mode is 266.02 seconds, versus 257.99 seconds when translating interactively.

5.2.1 Quality evaluation of the post-edited texts

In order to assess the quality of the translations, instead of asking volunteers to review the translations, we used the already human translated references from the WMT12[1] translation task from where we took the source texts. Using the professional translations as the reference, and the translation of this study as the hypothesis, we calculated the Error Rate per Editor and Document (Table 5.4).

Table 5.4: Translation Error Rate (TER) per Editor and Document

	Document	Mode	TER
Editor 1	1	ITP	0.44164 (140/317)
	2	PE	0.44681 (168/376)
Editor 2	1	ITP	0.36908 (117/317)
	2	PE	0.37500 (141/376)
Editor 3	1	PE	0.39432 (125/317)
	2	ITP	0.43085 (162/376)
Editor 4	1	PE	0.41009 (130/317)
	2	ITP	0.43351 (163/376)
Editor 5	1	ITP	0.43531 (138/317)
	2	PE	0.40691 (153/376)
Editor 6	1	PE	0.46372 (147/317)
	2	ITP	0.42021 (158/376)

[1] http://www.statmt.org/wmt12/ Seventh Workshop on Statistical Machine Translation in Quebec, Canada. Accessed April 19, 2014

As mentioned in Section 2.1.1, automatic evaluation correlates with human evaluation of translations. On average, as shown in Table 5.5, over all Documents the editors produce translations of similar quality using either edit mode (Interactive Translation vs Post Edit), with PE having slightly lower TER score. Lower TER indicates less changes in the original text (the hypothesis), therefore better quality. In this study, both TER scores are relatively high, which can be explained by the fact that the reference translations were composed by professional translators from scratch, without the help of Machine Translation. This means that the translation style is freer; the quality might be similar, but it makes automatic assessment more difficult. In future studies, a better approach for automatic evaluation would be to use post-edited texts from the same Machine Translated output as reference. For our purpose, that we want to assess the relative quality between the two editing modes, the approach used here already gives us an indication.

Table 5.5: Average (non normalized) TER per document and edit mode

Edit Mode	TER
ITP	0.42176
PE	0.41614

5.2.3. User Feedback

Analysing user activity logs is very important as it gives us objective information about the underlying translation process. Asking for user feedback is equally necessary in a user study, as it gives a way of evaluating the subjective user satisfaction.

In this study, user feedback was elicited in the form of questionnaires. After completing both translation tasks, the participants were asked to complete a short survey regarding the two translation modes (PE and ITP) and compare them.

First, as the participants are non-professional translators, we asked **general questions** regarding their language skills and their familiarity with Machine Translation:

1. Is English your first foreign language?
   Only one responded negatively

2. If not, which language(s) did you learn before English (and Spanish, of course)?
   The same participant responded that he had learned Italian first

3. How often do you use Machine Translation, like Google Translate? (Very often-Never)
   100% “Often”

Then we asked more **specific questions regarding the two editing modes** and the source texts:

4. Which interface enabled you to translate sentences more quickly? (PE vs ITP)
   83.33% Interactive Suggestions (ITP)
   16.67% Simple Editing of Machine Translation output (PE)

5. Were the two texts equally easy/difficult to translate?
   50% Yes
   16.67% No, the first text contained easier sentences
   33.33% No, the second text contained easier sentences

6. How useful were the initial translations (Machine Translations)?
   50% Very useful
   50% Useful

7. Would you have preferred to translate without any help from Machine Translation (in PE mode)?
   16.67% Yes
   83.33% No

8. How useful were the interactive suggestions (auto completions)?
   33.33% Very useful
   66.67% Useful

9. Comments regarding interactive suggestions (optional)

   Some of the most relevant comments are:

   1. "[The suggestions] were quite useful to find the correct way to start translating, or to find a more suitable word for the sentence"
   2. "The translations were pretty good, but I think there's still a **problem with Machine Translation between translation and interpretation**. Sometimes an accurate translation is not what you really mean or the way it's really said in Spanish, in this case. However, it's been **one of the best systems I have ever tried so far**, I didn't have to change many words in most of the cases, and in the second mode (ITP mode) sometimes I just pressed tab all the time. Still, you have to get used to it, as when you are translating something, you look at the text in English, thinking how to do it and then writing, plus **now check the suggestions, which can be good but different from what you are thinking**... **gets better after several tries, though**."

And about the Interactive Translation tool which was our point of interest:

10. How often did you accept the interactive suggestions? (Never-Rarely-Often-Always)
    16.67% Rarely
    83.33% Often

11. How often did you use "tab" to accept the interactive suggestions?
    16.67% Rarely

83.33% Often

12. How often did the tool give a correct suggestion, but you kept typing it instead of accepting it because you thought it would be faster/easier?
    16.67% Rarely
    83.33% Often
13. How true are the following statements: (Strongly disagree-Strongly agree)
    a. I was faster when using using the interactive translation (the auto completions)
        16.67% Disagree
        50.00% Agree
        33.33% Strongly agree

    b. I was faster when editing the machine translation output without the suggestions
        83.33% Disagree
        16.67% Agree

    c. I would have been faster if I had translated without any assistance from Machine Translation (without any Spanish text)
        33.33% Strongly disagree
        50% Disagree
        16.67% Agree

    d. I felt more confident about my translation because of the initial (non interactive) suggestions
        16.67% Strongly disagree
        33.33% Disagree
        33.33% Agree
        16.67% Strongly agree

    e. I felt more confident about my translation because of the interactive suggestions (autocompletions)
        16.67% Disagree
        50.00% Agree
        33.33% Strongly Agree

    f. I would have felt more confident about my translation if there was no assistance from Machine Translated text
        50.00% Strongly disagree
        50.00% Disagree

g. The autocompletions led me to a translation that was not correct, and I had to go back and make many edits
   33.33% Disagree
   66.67% Agree

h. The autocompletions helped me think of an appropriate translation faster than I would have on my own
   16.67% Disagree
   50.00% Agree
   33.33% Strongly agree

i. The Machine Translated text helped me think of an expression that I would have otherwise not remembered
   33.33% Disagree
   50.00% Agree
   16.67% Strongly agree

Most participants seem to be strongly in favour of the interactive translation mode, and only one editor (16.67%) believes that he would have translated better and faster without the help of Machine Translated output, and especially in interactive editing mode. When asked further to explain this dissatisfaction, this particular participant revealed that he was using an English keyboard, without the necessary accents, to translate the texts. Therefore, the suggestions had less chances of being correct, as the user prefix contained more errors; in Spanish, the same word can have different meaning if it is accented, for instance, *"tu"* means your, whereas *"tú"* means you. Therefore, the non-accented words were regarded as different words by the prediction algorithm and were not matched in the search graph. This participant was not thrown out as an outlier though, because this scenario could occur in real life, especially by a sloppy (and, even more, non-professional) translator.

5.3 Conclusions and Future Work

The results of this process study are promising, as they show that the editors were in favor of the Interactive Translation Prediction tool, according to the survey responses. However, their performance did not increase in terms of speed or quality.

For a future study, it would be more interesting to give emphasis on the optimal visualization option to display the autocompletion suggestions/predictions. As mentioned in Section 4, in TransType (Langlais et al., 2000) and Caitra (Koehn, 2009) typically 1-5 words are shown to the user, whereas in the CasMaCat project, there are several options:

i) the floating prediction that displays the next 3 tokens, which is used in this user study,

ii) displaying the whole prediction path in a light grey color, whereas the translation that the user has typed or accepted is displayed in black,
iii) same as the second option but the suggestions are displayed in black as well.

None of these displays has been proven to be the optimal one. However, it must be taken into consideration that the constant change of interactive suggestions, especially in a long sentence (e.g. with more than 20 tokens) probably increases the cognitive effort and, as a result, hinders the translation process. It is therefore interesting to compare these visualization options, and to determine what translators prefer and what boosts their productivity in terms of overall. To my knowledge no such study has been performed yet.

Chapter 6.
Conclusion

The Interactive Translation tool is an important advancement over simple post-editing of Machine Translated output. From our user study, the user satisfaction that derives from the questionnaire shows that the editors were strongly in favour of interactive editing. However, activity logs do not show an increase in translation speed; on the contrary, in some cases the completion time is lower in the case of post-editing. This implies that the interactive prediction tool needs slight improvements in order to truly aid translators via CAT tools and replace the conventional post-editing method.

However, an important factor of these results is the learning curve. It is interesting to compare the current results regarding translation speed with those of an extended user study (as in Sanchis-Trilles et al, 2014), where the editors are asked to work on more translation segments over a period of at least a week. This set up truly allows us to see whether the editors become more productive over time if trained even minimally.

Nevertheless, even with the current results, given that the user satisfaction is high, it is worth further developing Interactive Translation Prediction (ITP). More specifically, what needs further exploration is the improvement of the speed and accuracy of the ITP algorithm, as well as the optimal visualization option of the prediction to increase its usability. In order to optimize the usability potential, it would be interesting to run a longer user study using a few different ITP interfaces vs a post-edition mode as a baseline.

Of course, regarding future improvements, it is of major importance to improve the accuracy of the prediction itself, or to explore better programming solutions that will increase its speed. An increase in speed would allow the algorithm to exploring more features, such as the stem or levenshtein distance of the words matched, in order to find the optimal path in the search graph. Treating prediction as a Machine Learning problem did not lead to interesting conclusions, but there is still space to focus on the search graph and explore and test different features, such as the stem of the words or the suffix surrounding the last token that the user typed.

Usability consists of various additional things on top of the accuracy of the completion. For instance, the following matters need to be explored further regarding the visualization algorithm:

a) the number of suggestions presented to the user; only the best prediction, a list of suggestions in a dropdown menu, or a list of suggestion that will change using mouse actions
b) the optimal number of prediction tokens displayed; the three first as in this study, or more
c) the place where the suggestions are displayed; directly in the editing box or externally as in this study.

Both a) and b) have to do with the cognitive load of the user when presented with too many options to read. Presenting too much interchanging text will make it difficult for the translators to read on every keystroke, but displaying only 1-2 words is going to slow down the translation speed as well. Regarding the third point, the hypothesis is that users do not feel comfortable with the system interfering with their translations, even when it comes to suggesting changes; a more discreet way of display, such as the floating visualization tested here, is less invasive and therefore easier to accept.

Furthermore, the general layout of the CAT tool plays an important role in the perception of the usability of a given framework; it is important to follow basic page layout usability guidelines, so as not to confuse or tire the user with overcomplicated layouts. For instance (Travis, 2014), i) the relationship between buttons and their actions should be clear, ii) fonts are readable and consistent, iii) there is a good balance between whitespace and information density. The same holds for the Machine Translation quality, as discussed at sections 3 and 5: lower quality MT makes the application be perceived as less usable.

However, the key to increase usability is finding the optimal visualization options, namely the way that the interactive prediction is presented to the user, as mentioned above.

Bibliography

Alabau, V., Leiva, L. A., Ortiz-Martınez, D., & Casacuberta, F. (2012). User evaluation of interactive machine translation systems. *In Proc. EAMT* (pp. 20-23).

Alabau, V., Bonk, R., Buck, C., Carl, M., Casacuberta, F., García-Martínez, M., ... & Tsoukala, C. (2013). CASMACAT: An open source workbench for advanced computer aided translation. *The Prague Bulletin of Mathematical Linguistics,* 100, 101-112.

Alabau V., González-Rubio J., Ortiz-Martínez D., Sanchis-Trilles G., Casacuberta F., García-Martínez M., et al. (2014). Integrating Online and Active Learning in a Computer-Assisted Translation Workbench. *Proceedings of the Workshop on Interactive and Adaptive Machine Translation at the 11th conference of the Association for Machine Translation in the Americas (AMTA),* pp. 1-8

Alves, F., & Vale, D. C. (2009). Probing the unit of translation in time: aspects of the design and development of a web application for storing, annotating, and querying translation process data. *Across Languages and Cultures,* 10(2), 251-273.

Barrachina, S., Bender, O., Casacuberta, F., Civera, J., Cubel, E., Khadivi, S., Lagarda, A., Ney, H., Tomas, J., Vidal, E., and Vilar, J.-M. (2009). Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1):3–28.

Callison-Burch C., Fordyce C., Koehn P., Monz C., and Schroeder. J. (2007). (Meta-) Evaluation of Machine Translation, *In Proceedings of the ACL Workshop on Statistical Machine Translation,* pages 136-158, Prague, Czech Republic, June 2007.

Callison-Burch C., Fordyce C., Koehn P., Monz C., and Schroeder. J. (2008). Further Meta-evaluation of Machine Translation, *In Proceedings of the third ACL Workshop on Statistical Machine Translation*, pages 70-106, June 19-19, 2008, Columbus, Ohio

Callison-Burch C, Koehn P, Monz C, Post M, Soricut R, Specia L (2012). Findings of the 2012 workshop on statistical machine translation. *In: Proceedings of the Seventh Workshop on Statistical Machine Translation*, pp 10–51

Carl M., Dragsted B., Elming J., Hardt D. and Jakobsen A.L. (2011). The Process of Post-Editing: a Pilot Study. *In Proceedings of the 8th international NLPSC workshop. Special theme: Human-machine interaction in translation, pages 131-142*. Copenhagen Business School, 20-21 August 2011.

Carl, M. (2012). The CRITT TPR-DB 1.0: A Database for Empirical Human Translation Process Research. In S. O'Brien, M. Simard, & L. Specia (Eds.), *Proceedings of the AMTA 2012 Workshop on Post-Editing Technology and Practice (WPTP 2012)*. (pp. 9-18). Stroudsburg, PA: Association for Machine Translation in the Americas (AMTA).

Carl, M., Martinez, M. G., Mesa-Lao, B., Underwood, N., Keller, F., & Hill, R. L. (2013). Progress report on user interface studies, cognitive and user modelling. CASMACAT.

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

Desilets, A. (2009). Up close and personal with a translator – how translators really work. *In Machine Translation Summit XII*. Tutorial.

Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. *In Proceedings of the second international conference on Human Language Technology Research* (pp. 138-145). Morgan Kaufmann Publishers Inc.

Esteban, J., Lorenzo, J., Valderrábanos, A. S., & Lapalme, G. (2004). TransType2: an innovative computer-assisted translation system. *In Proceedings of the ACL 2004 on Interactive poster and demonstration sessions* (p. 1). Association for Computational Linguistics.

Foster G., Isabelle P., and Plamondon P. (1997). Target-text mediated interactive machine translation*, Machine Translation*, 12:175–194.

Foster, G., & Lapalme, G. (2002). Text prediction for translators. *Université de Montréal.*

Fraser, J. (1996). The translator investigated: Learning from translation process analysis. *The Translator 2*: 65-79

Galvez M. and Bhansali S. (2009). Translating the world's information with Google translator toolkit

Green, S., Cer, D., Manning, C. D. (2014). Phrasal: A toolkit for new directions in statistical machine translation. *In WMT 2014*

Gojun, A., & Fraser, A. (2012). Determining the placement of German verbs in English-to-German SMT. *In Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 726-735). Association for Computational Linguistics.

Isabelle, P., Church, K.W. (1991). New Tools for Human Translators, *Special Issue. Machine Translation* 12.1-2.

Jääskeläinen, R. (2010). Think-aloud protocol. *Handbook of Translation Studies*. Amsterdam: John Benjamins Publishing, 371-3.

Jakobsen, A. L., & Schou, L. (1999). Translog documentation. Probing the process in translation: methods and results. *Copenhagen Studies in Language Series*, 24, 1-36.

Jakobsen, A. L. (2003). Effect of think aloud on translation speed, revision and segmentation. *In Alves, F., editor, Triangulating Translation*, pages 69-96.

Koehn P., Hoang H., Birch A., Callison-Burch C., Federico M., Bertoldi N., Cowan B., Shen W., Moran C., Zens R., Dyer C., Bojar O., Constantin A., Herbst E. (2007). Moses: Open Source Toolkit for Statistical Machine Translation , *ACL 2007*

Koehn, P. (2009). A web-based interactive computer aided translation tool. *In Proceedings of the ACL Interactive Poster and Demonstration Sessions*

Koehn, P. (2009b). A process study of computer-aided translation, *Machine Translation Journal*, 2009, volume 23, number 4, pages 241-263

Koehn, P., Haddow, B. (2009). Interactive assistance to human translators using statistical machine translation methods. *In Machine Translation Summit XII*

Koehn, P. (2010). Enabling Monolingual Translators: Post-Editing vs. Options. *In Proceedings of NAACL HLT 2010: Human Language Technologies* – the 2010 annual conference of the North Americal Chapter of the Association for Computational Linguistics, June 2-4, Los Angeles, California, 537-545

Koehn, P. and Haddow, B. (2012). Towards effective use of training data in statistical machine translation. *In Proceedings of the Seventh Workshop on Statistical Machine Translation,* pages 363–367, Montreal, Canada. Association for Computational Linguistics.

Koehn, P., & Germann, U. (2014). The Impact of Machine Translation Quality on Human Post-editing. *In Proceedings of the Workshop on Humans and Computer-assisted Translation (HaCaT).* Gothenburg, Sweden (pp. 38-46).

Koehn, P., Tsoukala, C., & Saint-Amand, H. (2014). Refinements to Interactive Translation Prediction Based on Search Graphs. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics* (Volume 2: Short Papers)

Koponen, M., Aziz, W., Ramos, L., & Specia, L. (2012). Post-editinteilog time as a measure of cognitive effort. *In Proceedings of the AMTA Workshop on Postediting Technology and Practice* (pp. 11-20).

Koponen, M. (2012). Comparing human perceptions of post-editing effort with post-editing operations. *In Proceedings of the Seventh Workshop on Statistical Machine Translation* (pp. 181-190). Association for Computational Linguistics.

Kumaran A., Saravanan K. And Maurice S. (2008). wikiBABEL; Community creation of multilingual data. *In Babel Wiki workshop 2008: Cross-language communication*

Krings, H. P. (2001). Repairing Texts: Empirical Investigations of Machine Translation Post-Editing Processes (Geoffrey S. Koby, ed.), *The Kent State University Press*, Kent, Ohio & London

Langlais, P., Foster, G., and Lapalme, G. (2000). Transtype: a computer-aided translation typing system. *In Proceedings of the ANLP-NAACL 2000 Workshop on Embedded Machine Translation Systems.*

Langlais, P., & Lapalme, G. (2002). Trans Type: Development-Evaluation Cycles to Boost Translator's Productivity. *Machine Translation*, 17(2), 77-98.

LDC (2005). Linguistic data annotation specification: Assessment of fluency and adequacy in translations. *Revision 1.5.*

Levenshtein VI (1966). Binary codes capable of correcting deletions, insertions, and reversals, *Soviet Physics Doklady* 10: 707–10.

Lewis, C. H. (1982). Using the "Thinking Aloud" Method In Cognitive Interface Design *(Technical report). IBM*. RC-9265.)

Marcello F., Cattelan A., and Trombetti M. (2012). Measuring user productivity in machine translation enhanced computer assisted translation. *In the Proceedings of the Tenth Conference of the Association for Machine Translation in the Americas (AMTA)* URL http://www.mt-archive.info/AMTA-2012-Federico.pdf.

Nielsen J. (1993). Usability Engineering, *Morgan Kaufmann, Interactive technologies*

Och F. (2012). Re: Breaking down the language barrier—six years in [Web log comment]. *Retrieved from http://googleblog.blogspot.com/2012/04/ breaking-down-language-barriersix-years.html,* Accessed July 19, 2014

Ortiz-Martinez, D., Garcia-Varea, I., and Casacuberta, F. (2010). Online learning for interactive statistical machine translation. *In Proc. NAACL-HLT*, pages 546–554.

Papineni, K., Roukos, S.,Ward, T., Zhu, W. J. (2002). "BLEU: a method for automatic evaluation of machine translation". *ACL-2002:* 40th Annual meeting of the Association for Computational Linguistics. pp. 311–318.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research,* 12, 2825-2830.

Plitt, M. and Masselot F.(2010). A productivity test of statistical machine translation post editing in a typical localisation context. *Prague Bulletin of Mathematical Linguistics,* 93:7–16, 2010. URL http://ufal.mff.cuni.cz/pbml/93/art-plitt-masselot.pdf.

Sanchis-Trilles, G., Alabau, V., Buck, C., Carl, M., Casacuberta, F., García-Martínez, M., ... & Vidal, E. (2014). Interactive translation prediction versus conventional post-editing in practice: a study with the CasMaCat workbench. *Machine Translation*, 28(3-4), 217-235.

Sharmin, S., Špakov, O., Räihä, K.-J., and Jakobsen, A. L. (2008). Effects of time pressure and text complexity on translators' fixations. *In Proceedings of the Symposium on Eye Tracking Research and Applications.*

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., & Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. *In Proceedings of association for machine translation in the Americas* (pp. 223-231).

Travis, D. (2014). 38 page layout and visual design usability guidelines. *Retrieved December 1, 2014, from http://www.userfocus.co.uk/resources/layoutchecklist.htm*

Turchi, M., Negri, M., & Federico, M. (2013). Coping with the subjectivity of human judgements in MT quality estimation. *In 8th Workshop on Statistical Machine Translation* (p. 240).

Appendices

Appendix A.1 - CasMaCat and the prediction tool

This study was based on the CAT tool CasMaCat (Cognitive Analysis and Statistical Methods for Advanced Computer Aided Translation[1]), an open source workbench that offers advanced CAT functionality: post-editing machine translation (PE), interactive translation prediction (ITP), visualization of word alignment, extensive logging with replay mode, integration with eye trackers and e-pen.

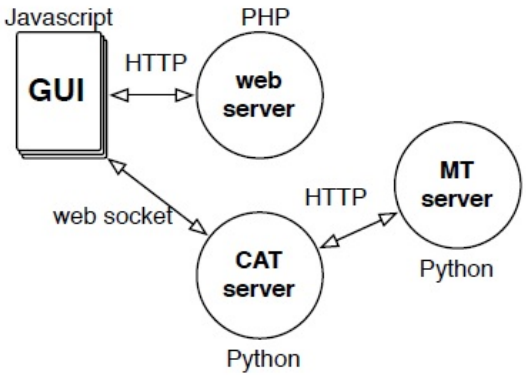
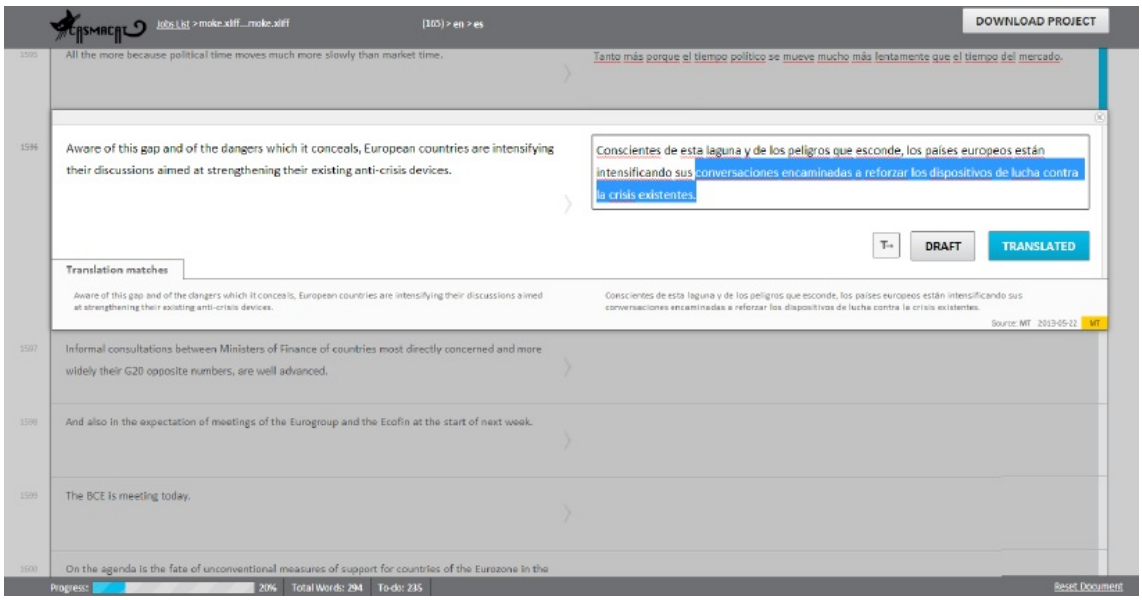


Figure A.1.1: Modules of the CasMaCat workbench: the GUI and the web server are web-based components. The CAT and MT server are independent and can be replaced by the MT engine of preference (e.g. Moses)

As seen on Figure A.1.1, CasMaCat consists of three parts: 1) the GUI, that is a combination of PHP scripts and JavaScript as it is web-based, 2) The MT engine (Moses) and a wrapper around the MT server and 3) the CAT server:

A1.1 GUI



[1] http://www.casmacat.eu

Figure A.1.2: Translation view of an English sentence (source language) to Spanish (target language), with simple post-editing configuration

A1.2 MT server

- Moses

Moses (Koehn et al., 2007) is an implementation of the statistical (or data-driven) approach to MT. It was developed at the University of Edinburgh and is one of the decoders used in CasMaCat

- Wrapper around the server

A server wrapper written in Python is used to handle all the requests to the decoder, as well as preprocessing of the data (such as tokenization, truecasing, normalization of punctuation) and post processing (detokenization, detruecasing). The wrapper was updated to return the search graph (in a JSON format) along with the best MT output.

A1.3 CAT server

CasMaCat is a web based tool, and therefore needs to handle multiple requests from the users. All the (Interactive) MT work is done on the CAT server. The CAT server calls the MT server wrapper to get the initial translation, as well as preprocessing and postprocessing information that is needed for the ITP binary. The prediction process is also handled by the CAT server.

The first integration of the prediction binary to the CasMaCat project involved only changes in the I/O. Namely, it was modified to read the search graph in a format given by the decoder (Figure A.1.3) instead of storing the states and transitions on a MySQL database, as it is done in the Caitra project (Koehn, 2009). Furthermore, in Caitra the algorithm generated a full prediction only after a full word was typed, whereas in CasMaCat the completion can start from the middle of a state as well (completion of partial output).

```
hyp,stack,back,score,transition,recombined,forward,fscore,covered-
start,covered-end,out
0,0,0,0,-1,140,-6.28898143768
6,1,0,-0.764266490936,-0.764266490936,-1,608,-5.84322500229,0,0,"me"
7,1,0,-0.813659667969,-0.813659667969,-1,629,-6.07921266556,0,0,"yo"
9,1,0,-0.980610847473,-0.980610847473,-1,733,-6.01157808304,0,0,"I"
15,1,0,-0.967086315155,-0.967086315155,-1,683,-6.06594848633,0,0,"he"
143,2,0,-1.69921016693,-1.69921016693,-1,619,-4.91799545288,0,1,"tengo"
144,2,0,-1.80319547653,-1.80319547653,-1,631,-5.02752161026,0,1,"debo"
254,2,0,-1.91500616074,-1.91500616074,-1,681,-4.99196481705,0,1,"I need"
352,3,140,-2.32472705841,-0.800611495972,-1,825,-3.96425437927,4,4,"mi"
362,3,140,-2.10974097252,-0.58562541008,-1,651,-4.55818295479,2,2,"a"
366,3,143,-2.53161716461,-0.832406997681,-1,625,-4.19932699203,4,4,"mi"
368,3,143,-2.65695762634,-0.957747459412,-1,879,-4.52334928513,4,4,"mis"
369,3,144,-2.71917057037,-0.915975093842,-1,881,-4.65649700165,4,4,"mi"
371,3,143,-2.30394053459,-0.604730367661,-1,780,-4.59933686256,2,2,"a"
```

Figure A.1.3: Example of the search graph as input (in a csv format) to the prediction binary. Brief explanation of the headers: *Hyp*: hypothesis (or state) number, *back*: previous state, *transition*: transition score, *recombined*: the state the current hypothesis was recombined with (or -1 if there was no recombination), *forward*: following state, *fscore*: forward path score, *covered-start* and *covered-end*: states of the source text that were covered by this transition, *out*: the output

Appendix B.1 – User study

Source (English) texts

Document 1

1. How to spot a lie
2. A glance at recent headlines indicates just how serious and pervasive deceit and lying are in daily life.
3. Lying has destroyed careers and convulsed countries.
4. And then again, no one who lived through it will ever forget the media circus President Bill Clinton unleashed by lying during his second term in office about his sexual involvement with Monica Lewinsky.
5. There have been instances where teachers have given students test answers in order to make themselves look good on their performance reviews.
6. Mentors who should be teaching the opposite are sending a message that lying and cheating are acceptable.
7. How much deceit do we encounter?
8. On a given day, studies show, you may be lied to anywhere from 10 to 200 times.
9. Now granted, many of those are white lies.
10. Another study showed that strangers lied three times within the first 10 minutes of meeting each other.
11. Detecting lies, or "lie spotting", is an essential skill for everyone to acquire, for both personal and professional reasons.
12. Liars do look you in the eye.
13. Don't conclude from this that liars are hard to spot and difficult to unmask.
14. A trained lie spotter can get to the truth by learning about statement structure, facial micro-expressions, question formation and timing.
15. Good liars are skilled at reading others well, putting them at ease, managing their own emotions and intuitively sensing how others perceive them.
16. How do you tell if someone is lying?
17. First, observe your subject's normal behavior.
18. This is called "baselining".

Document 2

19. It helps provide a reference point for measuring changes later.
20. Observe your subject's posture, laugh, vocal quality.
21. You'd better know if someone normally taps their foot all the time so you don't make unjust accusations when you see foot-tapping in the middle of the meeting.
22. Also, pay attention to your subject's language.
23. Deceptive individuals might also use distancing language: "I did not have sexual relations with that woman... Miss Lewinsky" or repeat a hard question in its entirety.
24. The most common verbal indicators are subtle.

25. Someone might use lots of "qualifying language" when answering a hard question: "well... to tell you the truth... as far as I know... to the best of my knowledge".
26. the democratic routine
27. Support for democracy in Latin America continues to edge up, as does backing for private enterprise.
28. Crime has become a bigger worry than unemployment.
29. And Brazil is seen as more influential than the United States across much of the region.
30. Two related things stand out in the results of this year's poll, taken in September and early October.
31. The first is Latin America's fairly sunny mood.
32. The second is the increasing stability of attitudes towards democracy and its core institutions.
33. Support for democracy has risen noticeably in several countries on the Pacific rim of South America (see table 1).
34. For example in Peru, where economic growth has averaged 6% a year since 2002, support for democracy has risen from a low of 30% in 2005 to 61% this year.
35. It also rose in Mexico, where the economy has recovered after suffering a big drop in output last year.
36. Some 31% say that either they or a close relative have been victims of crime over the past year, but that is down from 28% last year and is the lowest figure since 1995.

Machine Translated (MT) texts:

Document 1

1. Cómo reconocer una mentira
2. Un vistazo a titulares recientes indican cuán grave y generalizada, el engaño y la mentira son en la vida cotidiana.
3. La mentira ha destruido carreras y convulsionado países.
4. Y luego, una vez más, que nadie lo olvidará jamás a través de los medios de comunicación del Presidente Bill Clinton de circo desatado por mentir durante su segundo mandato sobre su relación sexual con Monica Lewinsky.
5. Ha habido casos en que los maestros han dado prueba de estudiantes respuestas a fin de aparentar buena sobre su desempeño exámenes.
6. Los mentores que deberían enseñar lo contrario están enviando un mensaje que mentir y engañar son aceptables.
7. Cuánto engaño lo encontramos?
8. En un día determinado, los estudios muestran, puede estar mintió a cualquier parte de 10 a 200 veces.
9. Ahora sentado, muchas de esas son mentiras piadosas.

10. Otro estudio mostró que extraños mintió tres veces en los primeros 10 minutos de sesión mutuamente.
11. La detección de mentiras, o "mentira" de puntería, es una habilidad esencial para todos a adquirir, tanto por razones personales y profesionales.
12. Mentirosos no les miren a los ojos.
13. No concluir que mentirosos son difíciles de detectar y difícil desenmascarar.
14. Una mentira capacitados spotter puede llegar a la verdad aprendiendo sobre declaración estructura, micro-expresiones faciales, cuestión formación y oportunidad.
15. Buenos mentirosos son expertos en lectura y otros, poniéndolas a gusto, la gestión de sus propias emociones y teleobservación intuitivamente perciben cómo otros.
16. ¿Cómo saber si alguien está mintiendo?
17. En primer lugar, observar su comportamiento normal del sujeto.
18. Esto se llama "determinar la capacidad de vigilancia".

Document 2

19. Ayuda a proporcionar un punto de referencia para medir los cambios más tarde.
20. Observar su postura del sujeto, reír, calidad vocal.
21. Deberías mejor saber si alguien normalmente sus grifos de pie todo el tiempo para no hacer acusaciones injustas pie cuando ve las escuchas en medio de la reunión.
22. Asimismo, prestar atención a su idioma del sujeto.
23. También podría utilizar personas engañosa distanciamiento language: "no tuve relaciones sexuales con esa mujer... Miss Lewinsky" o repetir una pregunta difícil en su totalidad.
24. Los indicadores verbales más comunes son sutiles.
25. Alguien podría utilizar un montón de "expresiones condicionales" al responder a un duro question: "bueno... para decirle la verdad... por lo que yo sé... a lo mejor de mi conocimiento".
26. La rutina democrática
27. El apoyo a la democracia en América Latina sigue borde, así como respaldo a la empresa privada.
28. La delincuencia se ha convertido en una mayor preocupación que el desempleo.
29. Y Brasil es visto como más influyentes que los Estados Unidos en gran parte de la región.
30. Dos cosas destacan en los resultados de la encuesta de este año, adoptada en septiembre y principios de octubre.
31. La primera es América Latina bastante soleado estado de ánimo.
32. El segundo es el aumento de la estabilidad de las actitudes hacia la democracia y sus instituciones básicas.
33. El apoyo a la democracia ha aumentado notablemente en varios países de la cuenca del Pacífico de América del Sur (see cuadro 1).
34. Por ejemplo, en Perú, donde el crecimiento económico ha promediado 6% un año desde 2002, el apoyo a la democracia ha aumentado entre un mínimo de 30% en 2005 a 61% este año.
35. También aumentó en México, donde la economía se ha recuperado después de sufrir una caída en la producción del año pasado.
36. Algunos 31% decir que ellos o un pariente cercano han sido víctimas de la delincuencia durante el año pasado, pero que se ha reducido desde 28% el año pasado y es la cifra más baja desde 1995.