D2.3: Final Report on Interactive Translation Prediction

Germán Sanchis-Trilles, Vicent Alabau, Francisco Casacuberta, José-Miguel Benedí, Jesús González-Rubio, Daniel Ortiz-Martínez, Philipp Koehn

Distribution: Public

CASMACAT
Cognitive Analysis and Statistical Methods
for Advanced Computer Aided Translation

ICT Project 287576 Deliverable D2.3

Project funded by the European Community
under the Seventh Framework Programme for
Research and Technological Development.

Project ref no.	ICT-287576
Project acronym	CASMACAT
Project full title	Cognitive Analysis and Statistical Methods for Advanced Computer Aided Translation
Instrument	STREP
Thematic Priority	ICT-2011.4.2 Language Technologies
Start date / duration	01 November 2011 / 36 Months

Distribution	Public
Contractual date of delivery	October 31, 2014
Actual date of delivery	January 7, 2015
Date of last update	January 7, 2015
Deliverable number	D2.3
Deliverable title	Final Report on Interactive Translation Prediction
Type	Report
Status & version	Final
Number of pages	23
Contributing WP(s)	WP2
WP / Task responsible	UPVLC
Other contributors	CS
Internal reviewer	Michael Carl
Author(s)	Germán Sanchis-Trilles, Vicent Alabau, Francisco Casacuberta, José-Miguel Benedí, Jesús González-Rubio, Daniel Ortiz-Martínez, Philipp Koehn
EC project officer	Aleksandra Wesolowska
Keywords	

The partners in CASMACAT are:

University of Edinburgh (UEDIN)
Copenhagen Business School (CBS)
Universitat Politècnica de València (UPVLC)
Celer Soluciones (CS)

For copies of reports, updates on project activities and other CASMACAT related information, contact:

The CASMACAT Project Co-ordinator
Philipp Koehn, University of Edinburgh
10 Crichton Street, Edinburgh, EH8 9AB, United Kingdom
pkoehn@inf.ed.ac.uk
Phone +44 (131) 650-8287 - Fax +44 (131) 650-6626

Copies of reports and other material can also be accessed via the project's homepage: http://www.casmacat.eu/

© 2014, The Individual Authors
No part of this document may be reproduced or transmitted in any form, or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission from the copyright owner.

1 Executive Summary

This last deliverable concerning WP2 details the progress done in the last year of WP2 *Interactive Translation Prediction*, of the CASMACAT project. In addition, in the case of the tasks that were already closed before the beginning of this last year, a short summary of the work done in the previous years will also be provided. Officially, only one task of this work-package was still active in this last year, even though ongoing work was still performed on some of the closed tasks. Since this WP focused on the basic aspects of translation prediction, the work performed implies in some cases basic research, including the development and implementation of new search strategies, novel ways of user interaction, or less conventional SMT models being applied to ITP. The main goal of this work-package is to supply the final user with a more flexible and efficient assistance system, be it by means of multi-modality, by introducing novel ways of interacting with the system, or by improving the quality of the translation completions provided by the underlying ITP system.

During this third year, work was performed in tasks 2.1, 2.2, 2.3, and 2.4:

Task 2.1 *Search and Machine Learning Criteria for Prediction* (month 1-18)
With this task already closed in the second year, the corresponding Section will briefly review the work performed during the first two years of the project. This work focused both on proposing a theoretically optimal prediction algorithm in replacement of the traditional maximum-a-posteriori approach, and on providing the CASMACAT prototype with error-correcting technology.

Task 2.2 *Multi-modality in Interactive Translation Prediction*(month 6-24)
Having this task finished by the end of the second year, some effort was still devoted during this last year in order to close the task completely, provide the CASMACAT prototype with a functional e-pen interface, and evaluate the e-pen interaction during the final field test. Such interaction capabilities were developed during the first two years of the project, dealing both with character and gesture recognition.

Task 2.3 *Prediction from active interaction* (month 13-36)
The work performed during the first year of activity of this task (months 13-24) aimed at aiding the human translators to spot faster which parts of the suggested automatic translation were prone to contain errors. To do this, confidence measures were used within the CASMACAT workbench to highlight potential translation errors. The comments of the users in the second field trial revealed that even though they seemed to appreciate such tool, its performance was bellow their expectations. In parallel, the closely related *Task 4.2: Active learning in ITP* (ended in month 24) showed very promising preliminary results worth to be further explored. Given this considerations, we decided to explore the most promising direction and use the effort allocated this year for Task 2.3 to conduct a complete user study of the active learning techniques developed in Task 4.2.

Task 2.4 *Prediction from Parse Forest* (month 6-18) With this task already closed in the second year, the corresponding Section will briefly review the work performed during the first two years of the project, and present some refinements. The work focused on producing a fully-functional grammar-based ITP system. Even though not tested in field trials, the laboratory results obtained were very encouraging, and have been published in an important international conference.

Task 2.5 *New SMT Models for Interactive Translation Prediction* (month 1-12)
With this task already closed in the second year, the corresponding Section will briefly review the work performed during the first two years of the project. Such work focused on researching two different novel SMT models, namely a phrase-based hidden semi-Markov model and a phrase-based finite state model. Even though such models provided appealing results in constrained tasks, they were not able to perform better than a fully-fledged SMT system trained with Moses in a general setting. For this reason, these models were not implemented in an ITP setting.

Contents

2 Background

Despite multiple and important advances obtained so far in the field of Statistical Machine Translation (SMT), current machine translation (MT) systems are in many cases not able to produce ready-to-use texts [1, 8]. Indeed, MT systems usually require human post-editing in order to achieve high-quality translations.

One way of taking advantage of MT systems is to interactively combine them with the knowledge of a human translator, constituting the Interactive Translation Paradigm (ITP) paradigm. This ITP paradigm can be considered a special type of the so-called computer-assisted translation (CAT) paradigm [12]. The main difference between ITP and traditional post-editing is that, in ITP, the system takes into account each interaction of the user, and attempts to build an improved translation hypothesis by completing the prefix validated by the user. Typical implementations of ITP systems are based on the generation of word translation graphs. During the interactive translation process of a given source sentence, the system makes use of the word graph generated for that sentence in order to complete the prefixes accepted by the human translator. Specifically, the system finds the best path in the word graph which is compatible with the user prefix.

The main advantages of word graph-based ITP systems is their efficiency in terms of the time cost per each interaction. This is due to the fact that the word graph is generated only once at the beginning of the interactive translation process of a given source sentence, and the suffixes required in ITP can be obtained by incrementally processing this word graph.

A common problem in ITP arises when the user sets a prefix which cannot be generated by the statistical models. Under these circumstances, the suffix cannot be appropriately generated, since the system is unable to generate translations that are compatible with the prefix validated by the user. In those ITP systems that use word graphs to generate the suffixes, the common procedure to face this problem is to perform a tolerant search in the word graph. This tolerant search uses the well known concept of Levenshtein distance in order to obtain the most similar string for the given prefix (see [19] for more details). These error-correcting techniques, although they are not included in the statistical formulation of the ITP process, are crucial to ensure that the suffixes completing the prefixes given by the user can be generated.

In this work, an alternative formalisation of the ITP framework which includes stochastic error-correction models in its statistical formulation is proposed. The proposed technique relies on word graphs to generate the suffixes required in ITP.

2.1 Statistical Machine Translation

The statistical approach to MT formalises the problem of generating translations under a statistical point of view. More formally, given a source sentence $\mathbf{x} \equiv x_1 ... x_j ... x_J$ in the source language $\mathcal{X}$, we want to find its equivalent target sentence $\mathbf{y} \equiv y_1 ... y_i ... y_I$[1] in the target language $\mathcal{Y}$.

From the set of all possible sentences of the target language, we are interested in the one with the highest probability according to the following equation:

$$\hat{\mathbf{y}} = \arg\max_{I,\mathbf{y}} \{\mathrm{p}(\mathbf{y} \mid \mathbf{x})\} \tag{1}$$

where $\mathrm{p}(\mathbf{y} \mid \mathbf{x})$ represents the probability of translating $\mathbf{x}$ into $\mathbf{y}$.

The early works on SMT were based on the use of *generative models*. A generative model is a full probability model of all statistical variables that are required to randomly generating observable data. Generative models decompose $\mathrm{p}(\mathbf{y} \mid \mathbf{x})$ applying the Bayes decision rule. Taking into account that $\mathrm{p}(\mathbf{x})$ does not depend on $\mathbf{y}$ we arrive to the following expression [7]:

$$\hat{\mathbf{y}} = \arg\max_{I,\mathbf{y}} \{\mathrm{p}(\mathbf{y})\, \mathrm{p}(\mathbf{x} \mid \mathbf{y})\} \tag{2}$$

where: $\mathrm{p}(\mathbf{x})$ represents the probability of generating the target sentence, and $\mathrm{p}(\mathbf{x} \mid \mathbf{y})$ is the probability of generating $\mathbf{y}$ given $\mathbf{x}$. Since the real probability distributions $\mathrm{p}(\mathbf{x})$ and $\mathrm{p}(\mathbf{x} \mid \mathbf{y})$ are not known, they are approximated by means of parametric statistical models. Specifically, $\mathrm{p}(\mathbf{x})$ is modelled by means of a *language model*, and $\mathrm{p}(\mathbf{x} \mid \mathbf{y})$ is modelled by means of a *translation model*. Current MT systems are based on the use of *phrase-based models* [13] as translation models. Typically, the values of the parameters of such statistical models are obtained by means of the well-known *maximum-likelihood* estimation method.

[1] x_j and y_i note the i'th word and the j'th word of the sentences $\mathbf{x}$ and $\mathbf{y}$ respectively.

More recently, alternative formalisations have been proposed. Such formalisations are based on the direct modelling of the posterior probability $\mathrm{p}(\mathbf{y} \mid \mathbf{x})$, replacing the generative models by *discriminative models*. Log-linear models use a set of feature functions $h_m(\mathbf{x}, \mathbf{y})$ each one with its corresponding weight λ_m:

$$\hat{\mathbf{y}} = \arg\max_{I,\mathbf{y}} \left\{ \sum_{m=1}^{M} \lambda_m h_m(\mathbf{x}, \mathbf{y}) \right\} \tag{3}$$

The direct optimisation of the posterior probability in the Bayes decision rule is referred to as *discriminative training* [18]. Since the features of regular SMT log-linear models are usually implemented by means of generative models, discriminative training is applied here only to estimate the weights involved in the log-linear combination. This process is typically carried out by means of the *minimum error rate training* (MERT) algorithm [20].

2.2 Interactive Machine Translation

As was already mentioned in previous section, the ITP framework constitutes an alternative to fully automatic MT systems in which the MT system and its user collaborate to generate correct translations. These correct translations are generated in a series of interactions between the ITP system and its user. Specifically, at each interaction of the ITP process, the ITP system generates a translation of the source sentence which can be partially or completely accepted and corrected by the user of the ITP system. Each partially corrected text segment (referred to from now on as prefix), is then used by the SMT system as additional information to generate better translation suggestions.

More formally, in the ITP scenario we have to find an extension $\mathbf{y}_s$ for a prefix $\mathbf{y}_p$ given by the user:

$$\hat{\mathbf{y}}_s = \arg\max_{\mathbf{y}_s} \left\{ \mathrm{p}(\mathbf{y}_s \mid \mathbf{x}, \mathbf{y}_p) \right\} \tag{4}$$

Applying the Bayes rule, we arrive at the following expression:

$$\hat{\mathbf{y}}_s = \arg\max_{\mathbf{y}_s} \left\{ \mathrm{p}(\mathbf{y}_s \mid \mathbf{y}_p)\, \mathrm{p}(\mathbf{x} \mid \mathbf{y}_p, \mathbf{y}_s) \right\} \tag{5}$$

where the term $\mathrm{p}(\mathbf{y}_p)$ has been dropped since it does not depend on $\mathbf{y}_s$.

Thus, the search is restricted to those sentences $\mathbf{y}$ which contain $\mathbf{y}_p$ as prefix. It is also worth mentioning that the similarities between Equation (5) and Equation (2) (note that $\mathbf{y}_p\mathbf{y}_s \equiv \mathbf{y}$) allow us to use the same models if the search procedures are adequately modified [6, 5].

3 Task 1: Search and machine learning criteria for prediction

3.1 Optimal decision rule for sequential interactive structured prediction

3.1.1 Introduction

Traditionally, ITP systems have used a *maximum-a-posteriori* (MAP) approach to the problem of searching for a good suffix (Equation 5). MAP is known to be an optimal decision rule when our goal is to obtain a perfect suffix, i.e., a suffix that perfectly matches the user's expectations. Unfortunately, current state-of-the-art technologies incur in substantial errors when trying to compute a perfect suffix, mainly due to practical approximations assumed in order to reduce the computational complexity. For this reason, ITP does not aim to produce perfect automatic translations but to allow the user to obtain the desired output with less editing effort, and thus, the decision rule must be adjusted accordingly.

3.1.2 Development

In [2] we derive an optimal decision rule for the suffix search problem that follows the Bayes decision theory. The resulting decision rule works by appending from left-to-right and word-by-word the most likely word that continues the prefix. Suppose that we have a prefix of length i, and we are predicting the word in the k-th position, then, the optimum decision rule in [2] states that:

$$\hat{y}_k = \arg\max_{y_k} \sum_{\mathbf{y}'_s} \mathrm{p}(y_k \cdot \mathbf{y}'_s | \mathbf{x}, \mathbf{y}_p \cdot \hat{y}_{i+1}^{k-1}) \tag{6}$$

where $\cdot$ is the concatenation operator. y_k is the word that is being predicted, and $\hat{y}_{i+1}^{k-1}$ is the sequence of words starting after $\mathbf{y}_p$ and before y_k that have been predicted in this iteration, but before word y_k. It should be noted that the probability in Equation 7, $\mathrm{p}(y_k \cdot \mathbf{y}'_s | \mathbf{x}, \mathbf{y}_p \cdot \hat{y}_{i+1}^{k-1})$, is an instantiation of the probability in Equation 4. Furthermore, also note that the complete prediction of the suffix $\mathbf{y}_s$ is obtained by applying Equation 7 repeatedly by increasing k until the end of sentences is reached.

Thus, the probability of the next word can be obtained by summing up over all the possible suffixes starting with that word. However, the number of suffixes can be exponential in number and, hence, it would be imposible to enumerate by the MT search engine in a reasonable amount of time. Consequently, we developed a practical and efficient algorithm for these equations that operates on word graphs (WG) [2]. In addition, we showed in D2.2 that WGs can be transformed to further improve decoding space and time performance by determinizing and minimizing the unweighted WG and, then, attaching the computed posterior probabilities. As a result, the suffixes in Equation 7 can be marginalized out resulting in the following equation:

$$\hat{y}_k = \arg\max_{y_k} \mathrm{p}(y_k | \mathbf{x}, \mathbf{y}_p \cdot \hat{y}_{i+1}^{k-1}) \tag{7}$$

which is, furhermore, consistent with findings in [21] Additionally, we did a small experiment in which the resulting WGs were reduced to $\simeq 20\%$ of the original number of states and to $\simeq 50\%$ of the original number of arcs, which seems a huge saving. In addition, the transformation is very fast, taking less than $500ms$ in most of the cases.

Next, we proved that MAP can be seen as a Viterbi-like approximation to Equation 7 where the summation over the suffixes is replaced by a maximum. Therefore, it is expected that if the maximum dominates over the summation, then MAP and the optimal approach obtain the same result. We tried with several corpora in different NLP problems and it seems that the maximum often dominates over the summation, rendering the Viterbi-like approach almost as performant as the optimal approach. However, another result is that MAP is not an *admissible decision rule*[2]. Thus, given that we have efficient algorithms for the optimal approach, it should be used in any case.

3.1.3 Conclusions

The empirical evaluation has shown that the optimum decision rule can be obtained efficiently from word graphs. In addition, we studied how to transform the optimum decision rule based on word graphs into a greedy version of the algorithm, which can reduce the number of nodes to a fifth and halve the number of arcs. Given that the improvements were not decisive, and that a human translator would most likely not notice the difference, we decided that such improvements did not justify modifying the core of the ITP protocol of the CASMACAT workbench, possibly compromising future evaluations.

[2] an admissible decision rule is a rule for making a decision such that there isn't any other rule that is always *better* than it [9].

3.2 Prediction as a Machine Learning Problem

The work on this task was concluded in year 2, so will just quickly review it here. When exploring the search graph, we find many possible prefix matches (and hence optimal path completions) that differ by string edit distance and path probability. Instead of defining the optimal solution as the highest probability path among minimal edit distance matches, we may consider other evidence.

Given a prefix and a search graph, we can sample a set of possible suffixes $\{\mathbf{y}_s^1, \mathbf{y}_s^2, ...\mathbf{y}_s^n\}$, for which we have features such as

- $h_1(\mathbf{y}_s)$ = string edit distance
- $h_2(\mathbf{y}_s)$ = log of the path probability

We can add additional features, such as an indicator feature that checks if the last word of the prefix was matched in the graph.

Given this set of features, we define a linear model

$$\hat{\mathbf{y}}_s = \arg\max_{\mathbf{y}_s} \Big\{ \sum_i \lambda_i \; h_i(\mathbf{y}_s) \Big\} \tag{8}$$

The objective of the machine learning problem is now to find optimal weights $\Lambda = \{\lambda_1, ..., \lambda_n\}$.

Given a set of search graphs for input sentences, and user corrections (we used post-edits of sentences in our experiment), we can know the correct extension for each prefix, so we have supervised training data to learn a binary classifier.

However, our experiments have not show great improvements over the canonical approach, and the added computation cost make the approach impractical in a real world setting, so we decided not to integrate this method into the CASMACAT workbench.

3.3 ITP Based on Stochastic Error-Correction Models

3.3.1 Introduction

A common problem in ITP arises when the user sets a prefix which cannot be produced by the statistical models. This problem requires the introduction of specific techniques in the ITP systems to guarantee that the suffixes can be generated. Part of the work developed within Task 1 has been focused on tackling the above mentioned problems by means of the introduction of error-correcting techniques. Previous related works use the concept of edit distance, however, this notion is not integrated into the statistical formulation of the ITP process.

During the first year of the project, an alternative formalisation of the ITP framework which includes stochastic error-correction models in its statistical formulation was proposed. Regarding the work carried out during the second year, it was focused on refining the ITP statistical formalism, as well as on integrating and deploying the proposed techniques into the CASMACAT Workbench.

3.3.2 Development

The above mentioned alternative formalisation of the ITP framework which includes stochastic error-correction models in its statistical formulation is based on the use of probabilistic finite-state machines (PFSMs) (see for instance [23, 24]), and more specifically, on the first stochastic interpretation of edit distance that was described in [4]. In that work, PFSMs were used to model the transformations produced by a noisy channel in a given text string.

The starting point of our alternative ITP formalisation consists in solving the problem of finding the sentence $\mathbf{y}$ in the target language that, at the same time, better explains the source sentence $\mathbf{x}$ and the prefix given by the user $\mathbf{y}_p$. This problem can be formally stated as follows:

$$\hat{\mathbf{y}} = \arg\max_{I,\mathbf{y}}\{\mathrm{p}(\mathbf{y} \mid \mathbf{x}, \mathbf{y}_p)\} \tag{9}$$

Using the Bayes rule and after some simplifications we can write:

$$\hat{\mathbf{y}} = \arg\max_{I,\mathbf{y}}\{\mathrm{p}(\mathbf{y})\,\mathrm{p}(\mathbf{x}, \mathbf{y}_p \mid \mathbf{y})\} \tag{10}$$

Now we make the following naive Bayes assumption: given the hypothesised target string $\mathbf{y}$, the strings $\mathbf{x}$ and $\mathbf{y}_p$ are considered statistically independent. Thus, we obtain the following expression:

$$\hat{\mathbf{y}} = \arg\max_{I,\mathbf{y}}\{\mathrm{p}(\mathbf{y})\,\mathrm{p}(\mathbf{x} \mid \mathbf{y})\,\mathrm{p}(\mathbf{y}_p \mid \mathbf{y})\} \tag{11}$$

In the previous equation the following terms can be found:

- $\mathrm{p}(\mathbf{y})$: measures the well-formedness of $\mathbf{y}$ as a sentence of the target language. This distribution can be approximated by means of a statistical language model.
- $\mathrm{p}(\mathbf{x} \mid \mathbf{y})$: measures the appropriateness of the sentence $\mathbf{x}$ as a possible translation of $\mathbf{y}$. This distribution can be approximated by means of a statistical translation model.
- $\mathrm{p}(\mathbf{y}_p \mid \mathbf{y})$: measures the compatibility of $\mathbf{y}$ with the user prefix $\mathbf{y}_p$. This distribution can be approximated by stochastic error-correction models based on PFSMs as it was mentioned at the beginning of this section.

It is worthy of note that the result of the maximisation given by Equation (11), $\hat{\mathbf{y}}$, may not contain the prefix $\mathbf{y}_p$ given by the user, since every possible target sentence $\mathbf{y}$ is compatible with the user prefix with a certain probability. Because of this, the problem defined by Equation (11) is not equivalent to the problem of finding the best suffix in ITP. To solve this problem, an additional assumption over the stochastic error-correction models is imposed. Specifically, they must be able to determine an *alignment* between a part of the target sentence $\mathbf{y}$ and the user prefix $\mathbf{y}_p$. The set of unaligned words of $\mathbf{y}$ in an appropriate order constitute the suffix required in ITP (see Deliverable 2.1 for a detailed explanation).

The above explained formalisation was also refined and combined with other translation technologies, such as hierarchical translation models (for more information see Section 6).

3.3.3 Conclusions

Laboratory experiments to test the performance of our proposed ITP formalisation with error-correction were carried out, measuring the effort required from the user to generate high-quality translations. Our system outperformed the results of those systems which are based on word graphs. In addition, the proposed formalisation was successfully integrated into the CASMACAT Workbench. The resulting software has been used in all the ITP experiments with real users made during the second and third field trials (for more details, see deliverables 6.2 and 6.3, respectively).

3.4 Refinements to Prediction Based on Search Graphs

While the standard approach of prediction based on search graphs works quite well, we spent some more time in the third year on refining the method. This resulted in a publication (attached at the end of this deliverable).

We require that a response has to be processed within 100 milliseconds, which is harder to fulfill if the prefix diverges too strongly from the search graph, as our experiments show (for instance a failure rate of 50% for 11 edits in a 35–40 sentence). We improve response times by threshold pruning of the search graph.

By (1) more permissive matching criteria, motivated by morphologically for case differences, (2) placing emphasis on matching the last word of the user prefix, and (3) dealing with predictions to partially typed words, we observe gains in both word prediction accuracy (+5.4%) and letter prediction accuracy (+9.3%). For details, please refer to Attachment A.

4 Task 2: Multi-modality in interactive translation prediction

4.1 Introduction

An alternative, perhaps more *natural* way of interacting with an ITP or post-editing system is by means of a touch screen or an electronic pen (e-pen). Handwritten words, characters and/or other editing gestures can be naturally and efficiently entered using an adequately designed e-pen interface. Although e-pen interaction may sound impractical for texts that need a large amount of corrections, there are a number of situations where it can actually be more comfortable and effective. First, it can be well suited for editing sentences with few errors, as it is the case of post-editing translation memory for sentences with high fuzzy matches, or the revision of human post-edited sentences. In these cases, it is worth reminding that e-pen is a just a *complementary* interface. The traditional keyboard and mouse are always available and the user can keep using at each moment the most comfortable interface, according to her preferences and/or the actual task being done. Second, it would allow to perform such tasks while commuting, travelling or sitting comfortably on the couch in the living room.

4.1.1 Multi-modal interactive text editing and reviewing

A main challenge which emerge when tackling the use of e-pen when post-editing MT and ITP is how to deal with the fact that on-line *hand-written text recognition* (HTR) systems do commit errors. As the user may weary of using a faulty e-pen system, HTR robustness needs to be improved. This is possible by taking advantage of information derived from the ITP process so as to achieve a system-user synergy which ultimately boosts both e-pen accuracy and usability.

The development of the CASMACAT e-pen interface follows closely the fundamental principles about multi-modal interaction established in [22]. In order to improve the accuracy of the e-pen recognizer, the basic equation for handwritten text recognition is extended by adding contextual information about the task at hand. Let $\mathbf{x}$ be the source text, $\mathbf{y}_p$ a user-validated output text prefix and $\mathbf{y}_s$ a system-suggested suffix (from a previous ITP step). Now let $\mathbf{t}$ be the user feedback in the form of *e-pen strokes*. Then, we can solve for the decoded word, $\hat{d}$, using the prefix, the (presumably wrong) suffix, the original source text, and the user feedback (on-line pen strokes), t:

$$\hat{d} \approx \arg\max_{d} \; \mathrm{p}(d \mid \mathbf{x}, \mathbf{y}_p, \mathbf{y}_s, \mathbf{t}) = \arg\max_{d} \; \mathrm{p}(\mathbf{t} \mid d)\, \mathrm{p}(d \mid \mathbf{x}, \mathbf{y}_p, \mathbf{y}_s) \qquad (12)$$

Equation 12 (right) is similar to the conventional statistical decoding equation for on-line HTR. The first term, $p(\mathbf{t} \mid d)$, is the *morphological likelihood*, which can be modelled, e.g., by character HMM models corresponding to the word(s) in d. The second term is a conditioned prior which can be provided by a special type of *language model*, constrained by information derived from the interaction process.

In [3] we leveraged this extra information to adapt the HTR language model to a specific translation context. In particular, we took advantage of the source sentence being translated, the portion of the translated sentence being supervised by the human, and the translation error to be amended. Empirical experimentation suggests that this is a valuable information to improve the robustness of the on-line HTR system achieving remarkable results. Also, adding a list of 10-best decodings led to reduce the errors by half.

Furthermore, the initial developments aimed at word-level interaction; i.e., the tokens to be corrected by means of e-pen strokes are single or multiple, full words. In [17] we aimed at developing more flexible interfaces where both character-level and multi-word (including incomplete words) interactive e-pen corrections are allowed.

4.2 Study of e-pen gestures for ITP

E-pen gestures perfectly complement on-line HTR since together they represent the digital analogy to a proof-reading process. Although there is already a 'de facto' standard for gestures for proof-reading the results with state-of-the-art gesture recognizers present high error rates. In [15] we presented a straightforward solution to incorporate text-editing gestures. Our approach provides disambiguation from handwritten text, excellent accuracy, and an algorithm that is trivial to implement and runs efficiently in any device. Furthermore, the gestures are easy to remember and reproduce. However, our gesture recognizer has a shortcoming: it provides a limited expressive power since it only allows for 8 possible commands operating on single words. In consequence, future efforts should go into finding new algorithms to recognize proof-reading marks with a high accuracy.

4.3 Conclusions

We have shown several works aiming at enabling an MT e-pen user interface. On the one hand, we have been able to define a set of gestures that cover an important set of editing operations. The recognizer is efficient and accurate. Furthermore, we have improved the robustness of the on-line HTR recognizer by leveraging contextual information. Finally, we have studied how to allow fine-grain editing operations by allowing character level and multi-word editing. To summarize, our work establishes sound bases for a feasible and complete design of an e-pen enabled MT system.

5 Task 3: Prediction from Active Interaction

Traditionally, ITP systems assume the user to systematically supervise each successive translation generated by the system, iteratively finding the point where the next translation error appears until the translation is correct. The goal of this task is to provide the human translator with a tool by means of which he can spot faster which parts of the translation suggestions provided by the underlying SMT system are prone to contain errors.

The proposed approach is based on the use of confidence measures (CM) to locate translation errors in which user attention should be focused. At each interaction, the workbench provides a new translation completion along with information about the reliability of each translated word. Then, those target words considered to be “incorrect”, i.e. unreliably translated words, are highlighted in a different color so the user can identify them easily. In our case, CM must be provided within an interactive environment where the user experience is crucial. Therefore, the ability to compute CM in real-time is a key characteristic to be taken into account.

A consequence of the time constraints inherent to interactive environments is that they specifically limit the complexity of the possible classification models. The temporal complexity of a CM is given by the complexity of computing the features plus the complexity of the classification model. The computation of the features usually involves a constant, or at most linear, time complexity given the input string. In contrast, the complexity of computing the quality score, except for the simplest classification models, usually involves more complex calculations that account for most of the complexity of the CM computation.

Taking these considerations into account, we decide to focus on computing a single feature as a direct estimator of the quality of the words [11]. Given the quality estimator, we can then classify each word as “correct” or “incorrect” depending on whether its quality score excess or not a certain word classification threshold τ_w. Note that other CM could surely provide better performance but their higher computational complexity forbids their use in an interactive environment. The particular details of the CM implemented can be found in deliverable D2.2.

During the first year of activity of this task, CM were integrated within the CASMACAT workbench so as to highlight potentially incorrect words in a different color. Such implementation was evaluated with real users in the second field trial, revealing that, even though users seem to appreciate such tool, the performance of the provided solution was below their expectations. The details of the implementation and the results of the field trial can also be found in deliverable D2.2.

In parallel, *Task 4.2: Active learning in ITP* obtained very promising preliminary results (see deliverables D4.2 and D4.3), showing an interesting potential to make a more efficient use of user effort within the ITP framework. Task 4.2 is closely related to Task 2.3 since both make use of similar ideas, although these are applied to achieve different goals. Particularly, both tasks make use of confidence measures but, while Task 2.3 is focused on the word level so to detect potential translation errors, Task 4.2 focuses on the sentence level to detect which of the automatic translations should be supervised by the user. The final goal of Task 4.2 is to achieve a (hopefully optimum) balance between the effort required from the users and the quality of the final translations.

Given these considerations, we decided that research effort would be more efficiently invested in further exploring Task 4.2 than Task 2.3. Therefore, during this third year of the project, we have invested the effort allocated for Task 2.3 (ending in month 36) to conduct complete user studies of the active learning techniques developed for Task 4.2. In these studies, we have compared the overall productivity of the CASMACAT workbench, as measured in translation quality per unit of user effort, when implementing 1) conventional ITP, 2) ITP with online learning (see Task 4.1 and deliverables D4.2 and D4.3), and 3) ITP with active leaning as developed for Task 4.2. The details of these user studies can be found in deliverable D6.3.

6 Task 4: Prediction from Parse Forest

This task was active only between months 6 and 18 of the project, we briefly sum up here the work performed at that time. For more details, please refer to deliverables D2.1 and D2.2.

6.1 Introduction

Although phrase-based translation models are still the most common approach within SMT, many grammar-based models have been proposed lately. Such models use hierarchical and syntactic information in order to improve translation quality. Grammar-based models have been widely used, specially for pairs of languages with strong differences in the syntactic structure, such as English-Chinese. However, the conventional ITP approach as described in [5] is designed to work with phrase-based models, ITP algorithms do not apply to grammar-based models. The aim of this task was thus to develop a fully-functional grammar-based ITP system.

6.2 Development

We worked on the foundations of Barrachina et al., [5], extending the methods described there so that they can be applied to grammar-based models. Specifically, we developed a formal ITP approach based on the use of hypergraphs that can be applied to both phrase-based and grammar-based models. Moreover, we also developed a stochastic error-correction model based on the Levenshtein distance [16] designed to address potential problems due to lack of prefix coverage. Our approach was described in an international conference article [10].

6.3 Conclusions

We evaluated the proposed ITP approach on two different translation tasks, see [10]. The comparative results against the conventional ITP approach described by Barrachina et al., [5] and a conventional post-edition approach showed that our ITP formalization for grammar-based SMT models indeed outperformed other approaches. Moreover, laboratory experiments indicated that large reductions in the human effort required to generate error-free translations could be achieved.

7 Task 5: New SMT Models for ITP

This task having been active only during the first year of the project, we briefly sum up here the work performed at that time. For more details, please refer to Deliverable D2.1.

7.1 Phrase-based hidden semi-Markov models

The bulk of the work presented in this subsection was performed during the first year of the CASMACAT project. The focus was to develop achieve state-of-the-art results with a hidden semi-Markov model with a strong theoretical foundation. The main idea behind the phrase-based hidden semi-Markov model (PBHSMM) is to model the conditional probability $p(\mathbf{x} \mid \mathbf{y})$ assuming that a monotonic translation process has been carried out from left to right in segments of words or phrases (for more detailes see Deliverable D2.1).

Even though the results obtained in terms or SMT evaluation metrics were competitive, and in certain circumstances even better than state-of-the-art phrase-based systems, work in this direction was not carried on to the final ITP prototype because of the high computational cost involved. This aspect is quite important whenever the PBHSMM is to be implemented within an interactive framework. In such a scenario, computational time is critical, since it is not acceptable to have the user waiting actively for the translation to be produced. Since the computational complexity is quite high, the improvements obtained in terms of translation quality might not justify the time required to produce the output, which might even render the system seemingly unresponsive. For this reason, this research direction was discontinued within the CASMACAT framework, and no results with ITP are available.

7.2 Finite-state models

Phrase-based finite-state models were built following a log-linear framework similar to that in Moses, although a monotone translation approach was adopted because of theoretical restrictions. Even though finite-state models were not in the initial planning of the project, recent de- velopments led to the idea that they could actually provide a complementary framework for providing the ITP engine with different SMT systems. For this reason, work on finite-state was performed during the first year of the project, with the purpose of determining whether such possibility was actually feasible.

Even though the results achieved by our finite-state log-linear approach to phrase-based SMT are comparable to those obtained by monotone Moses, the comparison with a fully-fleged, state- of-the-art SMT system with reordering would be less convincing. For this reason, this work was finally discarded for its application within an ITP framework.

References

[1] Nist 2006 machine translation evaluation official results. `http://www.itl.nist.gov/iad/mig/tests/mt/2006/doc/mt06eval_official_results.html`, November 2006.

[2] Vicent Alabau, Alberto Sanchis, and Francisco Casacuberta. On the optimal decision rule for sequential interactive structured prediction. *Pattern Recognition Letters*, 33(6):2226–2231, 2012.

[3] Vicent Alabau, Alberto Sanchis, and Francisco Casacuberta. Improving on-line handwritten recognition in interactive machine translation. *Pattern Recognition*, 2013. In press.

[4] L. R. Bahl and F. Jelinek. Decoding for channels with insertions, deletions, and substitutions wi th applications to speech recognition. *IEEE Transactions on Information Theory*, IT-21(4):404–411, 1975.

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

[6] O. Bender, S. Hasan, D. Vilar, R. Zens, and H. Ney. Comparison of generation strategies for interactive machine translation. In *Conference of the European Association for Machine Translation*, pages 33–40, Budapest, Hungary, May 2005.

[7] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993.

[8] C. Callison-Burch, C. Fordyce, P. Koehn, C. Monz, and J. Schroeder. (meta-) evaluation of machine translation. In *Proceedings of the ACL Workshop on Statistical Machine Translation*, pages 136–158, Prague, Czech Republic, June 2007.

[9] Y. Dodge, D. Cox, D. Commenges, A. Davison, P. Solomon, and S. Wilson. *The Oxford dictionary of statistical terms*. Oxford University Press, USA, 2006.

[10] Jesús González-Rubio, Daniel Ortiz-Martínez, José-Miguel Benedí, and Francisco Casacuberta. Interactive machine translation using hierarchical translation models. In *Proceedings of the conference on Empirical methods on natural language processing*, 2013.

[11] Jesús González-Rubio, Daniel Ortiz-Martínez, and Francisco Casacuberta. On the use of confidence measures within an interactive-predictive machine translation system. In *Proceedings of 14th Annual Conference of the European Association for Machine Translation*, 2010. http://www.mt-archive.info/EAMT-2010-Gonzalez-Rubio.pdf.

[12] P. Isabelle and K. Church. Special issue on new tools for human translators. *Machine Translation*, 12(1–2), 1997.

[13] P. Koehn, F. J. Och, and D. Marcu. Statistical phrase-based translation. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference*, pages 48–54, Edmonton, Canada, May 2003.

[14] Philipp Koehn, Chara Tsoukala, and Herve Saint-Amand. Refinements to interactive translation prediction based on search graphs. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 574–578, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[15] Luis A. Leiva, Vicent Alabau, and Enrique Vidal. Error-proof, high-performance, and context-aware gestures for interactive text edition. In *Proceedings of the 2013 annual conference extended abstracts on Human factors in computing systems (CHI EA)*, pages 1227–1232, 2013.

[16] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady.*, 10(8):707–710, February 1966.

[17] Daniel Martín-Albo, Verónica Romero, and Enrique Vidal. Interactive off-line handwritten text transcription using on-line handwritten text as feedback. In *International Conference on Document Analysis and Recognition (ICDAR)*, pages 1312–1316, 2013.

[18] H. Ney. On the probabilistic-interpretation of neural-network classifiers and discriminative training criteria. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):107–119, February 1995.

[19] F. J. Och, R. Zens, and H. Ney. Efficient search for interactive statistical machine translation. In *Tenth Conference of the European Chapter of the Association for Computational Linguistics*, pages 387–393, Budapest, Hungary, April 2003.

[20] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41th Annual Conference of the Associations for Computational Linguistics*, pages 160–167, Sapporo, Japan, July 2003.

[21] José Oncina. Optimum Algorithm to Minimize Human Interactions in Sequential Computer Assisted Pattern Recognition. *Pattern Recognition Letters*, 30(6):558–563, 2009.

[22] Alejandro H. Toselli, Enrique Vidal, and Francisco Casacuberta, editors. *Multimodal Interactive Pattern Recognition and Applications.* Springer, 1st edition edition, 2011. http://www.springer.com/computer/hci/book/978-0-85729-478-4.

[23] E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, and R. Carrasco. Probabilistic finite-state machines - part I. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1013–1025, 2005.

[24] E. Vidal, F. Thollard, C. de la Higuera, F. Casacuberta, and R. Carrasco. Probabilistic finite-state machines - part II. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1025–1039, 2005.

Attachment A

Task 2.1

Philipp Koehn, Chara Tsoukala, and Herve Saint-Amand.

Refinements to interactive translation prediction based on search graphs.

In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 574–578, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

Refinements to Interactive Translation Prediction Based on Search Graphs

Philipp Koehn[◇⋆], **Chara Tsoukala**[⋆] **and Herve Saint-Amand**[⋆]
[◇]Center for Speech and Language Processing, The Johns Hopkins University
[⋆]School of Informatics, University of Edinburgh
phi@jhu.edu, ctsoukal@inf.ed.ac.uk, hsamand@inf.ed.ac.uk

Abstract

We propose a number of refinements to the canonical approach to interactive translation prediction. By more permissive matching criteria, placing emphasis on matching the last word of the user prefix, and dealing with predictions to partially typed words, we observe gains in both word prediction accuracy (+5.4%) and letter prediction accuracy (+9.3%).

1 Introduction

As machine translation enters the workflow of professional translators, the exact nature of this human-computer interaction is currently an open challenge. Instead of tasking translators to post-edit the output of machine translation systems, a more interactive approach may be more fruitful.

One such idea is interactive translation prediction (Langlais et al., 2000b): While the user writes the translation for a sentence, the system makes suggestions for sequent words. If the user diverges from the suggestions, the system recalculates its prediction, and offers new suggestions. This input modality is familiar to anybody who has used auto-complete functions in text editors, cell phones, or web applications.

The technical challenge is to come up with a method that predicts words that the user will accept. The standard approach to this problem uses the search graph of the machine translation system. Such search graphs may be recomputed in a constraint decoding process restricted to the partial user input (called the *prefix*), but this is often too slow with big models and limited computing resources, so we use static word graphs.

The user prefix is matched against the search graph. If the user prefix cannot be found in the search graph, approximate string matching is used by finding a path with minimal string edit distance, i.e., a path in the graph with the minimal number of insertions, deletions and substitutions to match the user prefix.

This paper presents a number of refinements to extend this approach, by allowing more permissive matching criterion, placing emphasis on matching the last word of the user prefix, and dealing with predictions to partially typed words. We show improvements in word prediction accuracy from 56.1% to 60.5% and letter prediction accuracy from 75.2% to 84.5% on a publicly available benchmark (English-Spanish news translation).

2 Related Work

The interactive machine translation paradigm was first explored in the TransType and TransType2 projects (Langlais et al., 2000a; Foster et al., 2002; Bender et al., 2005; Barrachina et al., 2009). Given the computational cost and need for quick response time, most current word operates on search graphs (Och et al., 2003). Such search graphs can be efficiently represented and processed with finite state tools (Civera et al., 2004). More recently, the approach has been extended to SCFG-based translation models (González-Rubio et al., 2013).

There are several ways the sentence completion predictions can be presented to the user: showing the complete sentence prediction, only a few words, or multiple choices. User actions may be also extended to mouse actions to pinpoint the divergence from an acceptable translation (Sanchis-Trilles et al., 2008), or hand-writing (Alabau et al., 2011) and speech modalities (Cubel et al., 2009).

3 Properties of Core Algorithm

Our implementation of the core algorithm follows closely Koehn (2009). It is a dynamic programming solution that computes the minimal cost to reach each node in the search graph by matching parts of the user prefix. Cost is measured primarily in terms of string edit distance (number of deletions, insertions and substitutions), and secondary in terms of translation model score for the matched path in the graph. Search is done iteratively, with an increasing number of allowable edits.

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 574–578,
Baltimore, Maryland, USA, June 23-25 2014. ©2014 Association for Computational Linguistics

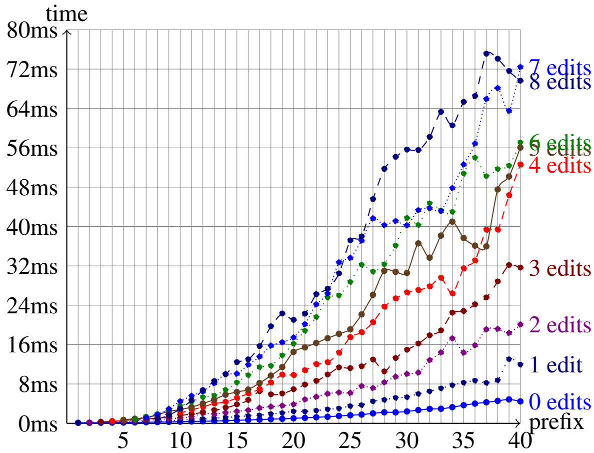

Figure 1: Average response time of baseline method based on length of the prefix and number of edits: The main bottleneck is the string edit distance between prefix and path.

3.1 Experimental Setup

Given the large number of proposed variations of the algorithm, we do not carry out user studies, but rather use a simulated setting. We predict translations that were crafted by manual post-editing of machine translation output. We also use the search graphs of the system that produced the original machine translation output.

Such data has been made available by the CASMACAT project[1]. In the project's first field trial[2], professional translators corrected machine translations of news stories from a competitive English–Spanish machine translation system (Koehn and Haddow, 2012). This test set consists of 24,444 word predictions and 141,662 letter predictions.

3.2 Prediction Speed

Since the interactive translation prediction process is used in an interactive setting where each key stroke of the user may trigger a new request, very fast response time is needed. According to standards in usability engineering

> *0.1 second is about the limit for having the user feel that the system is reacting instantaneously* (Nielsen, 1993).

So, this is the time limit we have to set ourselves to predict the next words of a translator.

What are the main factors that influence processing time in our core algorithm? See Figure 1 for an illustration. We plot processing time against the length of the user prefix and the string edit distance between the user prefix and the search graph. The graph clearly shows that the main slowdown in processing time occurs when the edit rate increases.

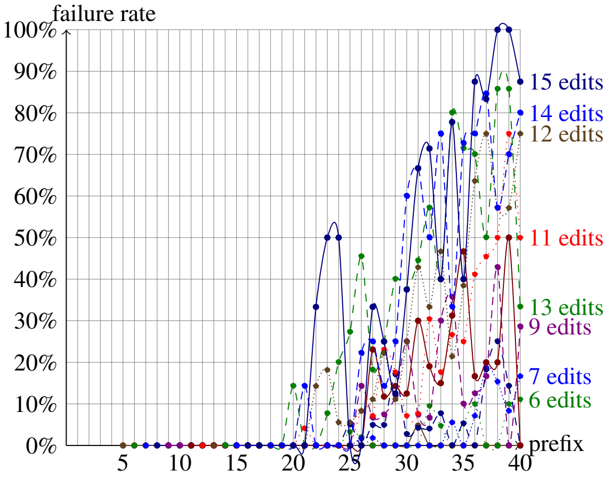

Figure 2: Ratio of prefix matching processes abandoned due to exceeding the 100ms time limit (showing only curves with a minimum of 5 edits).

To guarantee a response in 100ms, the algorithms aborts when this time is exceeded and relies on a prediction based on string edit distance against the best path in the graph. The larger the number of edits, the more often this occurs, as Figure 2 shows.

3.3 Accuracy

We are mainly interested in the accuracy of the method: How often does it predict a word that the user accepts? There is a trade-off between speed and accuracy.

One way we can balance this trade-off is by removing nodes from the search graph. By threshold pruning (Sanchis-Trilles and Ortiz-Martínez, 2014), we remove nodes from the search graph that are only part of paths that are worse than the best path by a specified score difference.

See Table 1 how the choice of the score difference threshold impacts failure rate and accuracy. A wider threshold has the potential to achieve better results (if we allows for up to 1 second of processing time), but with the constraint of 100ms response time, the optimum is with a threshold of 0.4. Wider thresholds lead to a higher failure rate, causing overall lower accuracy.

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

[2] http://www.casmacat.eu/uploads/Deliverables/d6.1.pdf

Threshold	100ms Max		1000ms Max	
	Acc.	Fail	Acc.	Fail
0.3	55.8%	4.5%	56.9%	0.0%
0.4	**56.1%**	**6.5%**	**58.0%**	**0.0%**
0.5	55.9%	9.0%	58.8%	0.0%
0.6	55.5%	11.6%	59.4%	0.0%
0.8	54.4%	17.1%	59.4%	0.0%
1.0	52.7%	21.7%	58.6%	6.5%

Table 1: Impact of threshold pruning on search accuracy and failure rate (i.e., failure to complete search in given time and resorting to matching against best translation).

4 Refinements

We now introduce a number of refinements over the core method. Given the constraints established in the previous section (maximum response time of 100ms, pruning threshold 0.4), we set out to improve accuracy.

4.1 Matching Last Word

The first idea is that it is more important to match the last word of the user prefix than having mismatches in earlier words. We attempt to find the last word in the predicted path either before or after the optimal matching position according to string edit distance.

We combine the matched path in the prefix with the optimal suffix, and search for the last user prefix word within a window. This means that we either move words from the suffix to the prefix or the other way around, without changing the overall string along the path.

Table 2 shows the impact on accuracy for different window sizes. While we expected some gains by checking for the word somewhere around the optimal position in the predicted path, we do see significant gains by not placing any restrictions to where the word can be found, except for a bias to less distant positions. For instance, examining a window of up to 3 words gives us a word prediction accuracy of 57.2% versus the 56.1% baseline. Finding the last word anywhere boosts performance to 59.1%.

The table also reports accuracy numbers when we allow the process to run up to 1 second — which is basically an exhaustive search but not practically useful. These numbers shed some light on why an unlimited window size in matching the last word helps: the gains come partially from the cases where the initial search fails. Finding the last user word anywhere in the machine translation output is a better fallback than computing optimal string edit distance. Analysis of the data suggests that gains mainly come from large length mismatches between user translation and machine translation, even in the case of first pass searches.

Window	100ms Max	1000ms Max
baseline	56.1%	58.0%
1 word	56.6%	58.4%
2 words	56.9%	58.6%
3 words	57.2%	58.9%
5 words	57.8%	59.3%
anywhere	59.1%	59.5%

Table 2: Search for the last prefix word in a window around the predicted position in the matched path.

Word Matching	100ms Max	1000ms Max
baseline	59.1%	59.5%
case-insensitive	58.7%	59.4%

Table 3: Search with case-insensitive word matching (say, *University* and *university*).

4.2 Case-Insensitive Matching

Some mismatches between words matter less than others. For instance, if the user prefix differs only in casing from the machine translation (say, *University* instead of *university*), then we may still want to treat that as a word match in our algorithm. However, as Table 3 shows, allowing case-insensitive matching leads to lower accuracy (58.7% vs. 59.1%).

A major reason is computational cost. The most inner loop in the algorithm compares words. This is optimized by representing words as integers. However, if we allow case-insensitive matching, this simple method does not work anymore. We do precompute approximate word matches and store matching words identifiers in a hash map, but still the ratio of searches that do not complete in 100ms increases from 6.5% to 9.7%. By extending the allowable time to 1 second, the accuracy gap is reduced to 0.1%.

4.3 Approximate Word Matching

When a word in the user translation differs from a word in the decoder search graph only by a few letters, then it should be considered a lesser error than substitutions of completely different words. Such word differences may be due to casing, morphological variants, or spelling inconsistencies.

We compute word dissimilarity by computing

Max. Dissimilarity	100ms Max.	1000ms Max.
baseline	59.1%	59.5%
30%	60.2%	61.0%
20%	60.4%	61.3%
10%	60.6%	61.5%

Table 4: Counting substitutions between similar words as half an error. Dissimilarity is measured as letter edit distance

Min Stem / Max Suffix	100ms	1000ms
baseline	59.1%	59.5%
4 / 3	59.4%	60.1%
3 / 3	59.5%	60.2%
2 / 3	59.5%	60.3%

Table 5: Counting substitutions between morphological variants as half an error. Morphological variance is approximated by requiring a minimum number of initial letters to match and a maximum of final letters to differ.

the ratio of letter edit operations to the length of the shorter word.[3] We now set a threshold for maximum dissimilarity, under which mismatched words are considered only half the edit cost of other edit operations.

Table 4 shows that we get significantly higher word prediction accuracy than with the baseline approach (up to 60.6% vs. 59.1%), and the best performance with a 10% threshold. We observe the same computational problem as in the previous section (about 9.2% first pass failures, vs. 6.5%), reflected in a higher accuracy gap for 100ms and 1000ms time limits.

4.4 Stemmed Matching

We suspected that the main benefit of approximate word matching is the better handling of morphological variants. In Spanish, this mainly constitutes itself as different word endings. Thus, we redefine our word dissimilarity measure by consider words similar, if they agree in at least a number of leading letters (presumably the stem), and may differ in at most a number of trailing letters (presumably the morpheme).

Table 5 shows that this is successful in increasing the word prediction rate (59.5% vs. 59.1%) but not as much as with the more general approximate word matching in the previous section (recall: 60.6%).

#	Method	Word Acc.	Letter Acc.
1	baseline	56.0%	75.2%
2	1+matching last word	59.0%	80.6%
3	2+case insensitive	58.7%	80.4%
4	2+dissimilarity 10%	60.5%	80.6%
5	2+stem 2/3	59.4%	80.5%
6	4+desperate	60.5%	84.5%

Table 6: Extending the approach to word completion. Impact of refinements of letter prediction accuracy with additional desperate word matching against the entire vocabulary.

5 Word Completion

Besides word prediction, word completion is also a useful feature in an interactive translation tool. When the machine translation system decides for *college* over *university*, but the user types the letter *u*, it should change its prediction.

To enable word completion in the canonical algorithm, we allow matching of the final user word (if not followed by a space character) as a prefix of any word as a zero cost operation. The predicted suffix that is returned to the user then starts with the remaining letters of the word in the path.

Table 6 shows that the refinements that helped sentence completion also benefit word completion. From a baseline accuracy of 75.2% correctly predicted letters, we reach up to 80.6%. Note that the baseline word prediction accuracy is slightly lower (56.0% vs. 56.1%) than in the previous experiments, since the previously correctly matched last word may be mistaken as the prefix of another word.

We add an additional refinement to this task: If the potentially incomplete final word of the user prefix cannot be found in the predicted path, then we explore the entire vocabulary from the unpruned search graph for completions. If multiple words match, the one with the highest path score is used. This *desperate* word completion method gives significant gains (84.5% over 80.6%).

6 Conclusion and Future Work

We observe most improvements by a focus on the last word of the user prefix and approximate word matching. This suggests that there may be additional gains by a stronger focus on the tail of the user prefix. Also, the findings from the time/productivity tradeoffs indicate that more time efficient algorithms and implementations should be explored.

[3] For instance, if a 6 letter word and a 4 letter word can be matched with two deletions and one substitution, then the dissimilarity score is $\frac{3}{4} = .75$.

Acknowledgements

This work was supported under the CASMACAT project (grant agreement N^{o} 287576) by the European Union 7th Framework Programme (FP7/2007-2013).

References

Alabau, V., Sanchis, A., and Casacuberta, F. (2011). Improving on-line handwritten recognition using translation models in multimodal interactive machine translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Techologies*, pages 389–394, Portland, Oregon, USA. Association for Computational Linguistics.

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

Bender, O., Hasan, S., Vilar, D., Zens, R., and Ney, H. (2005). Comparison of generation strategies for interactive machine translation. In *Proceedings of the 10th Conference of the European Association for Machine Translation (EAMT)*, Budapest.

Civera, J., Cubel, E., Lagarda, A. L., Picó, D., González, J., Vidal, E., Casacuberta, F., Vilar, J. M., and Barrachina, S. (2004). From machine translation to computer assisted translation using finite-state models. In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004*, pages 349–356, Barcelona, Spain. Association for Computational Linguistics.

Cubel, E., Khadivi, S., Lagarda, A., Ney, H., Toms, J., Vidal, E., and Vilar, J.-M. (2009). Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1).

Foster, G., Langlais, P., and Lapalme, G. (2002). User-friendly text prediction for translators. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 148–155, Philadelphia. Association for Computational Linguistics.

González-Rubio, J., Ortíz-Martinez, D., Benedí, J.-M., and Casacuberta, F. (2013). Interactive machine translation using hierarchical translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 244–254, Seattle, Washington, USA. Association for Computational Linguistics.

Koehn, P. (2009). A process study of computer-aided translation. *Machine Translation*, 23(4):241–263.

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.

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

Langlais, P., Foster, G., and Lapalme, G. (2000b). Unit completion for a computer-aided translation typing system. In *Proceedings of Annual Meeting of the North American Chapter of the Association of Computational Linguistics (NAACL)*.

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

Och, F. J., Zens, R., and Ney, H. (2003). Efficient search for interactive statistical machine translation. In *Proceedings of Meeting of the European Chapter of the Association of Computational Linguistics (EACL)*.

Sanchis-Trilles, G. and Ortiz-Martínez, D. (2014). Efficient wordgraph pruning for interactive translation prediction. In *Annual Conference of the European Association for Machine Translation (EAMT)*.

Sanchis-Trilles, G., Ortiz-Martínez, D., Civera, J., Casacuberta, F., Vidal, E., and Hoang, H. (2008). Improving interactive machine translation via mouse actions. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 485–494, Honolulu, Hawaii. Association for Computational Linguistics.