Comparison of monolingual and bilingual approach to predictive translation typing

Abstract

Our goal is to introduce and analyze simple text processing methods for enhancing predictive typing capabilities of computer assisted translation tools. In this paper we investigate the effect of improving two aspects of monolingual word based predictive dictionaries: instead of single standalone words our method uses sequences, and it can offer completions based on the content of the source language text. For evaluation multiple goodness metrics were analyzed and finally the number of keystrokes saved by the translator was chosen. A counter example is also presented to demonstrate that in our case coverage type metrics do not always correlate with keystroke saving, as opposed to the general belief in the domain of statistical machine translation.

1 Introduction

The ultimate research aim in the field of machine translation (MT) is to develop computer systems capable of providing high-quality translation without human interaction. Machine translation systems has not been able to reach this maturity yet, thus three basically different approaches have been designed to include a human translator's knowledge in order to improve the quality of the output. These ones are *post-edition*, *pre-edition* and *interactive predictive typing* (Allen, 2003; Brown and Nirenburg, 1990). In post-edition the output of the MT system is reviewed and corrected by the translator, while in pre-edition the user helps to disambiguate the source text for the MT system.

Our work is carried out in the context of *interactive predictive translation typing*. In this approach the communication medium between machine and translator is the translation text currently being typed (Foster et al., 1997). As the translator types the translation, the machine translation system continuously provides her with possible translation completions, which can be accepted or completely ignored. This way the control always remains in the translator's hand, while the computer system adapts to her, thus she can work the way she would normally do, but her productivity is possibly increased by help of the computer. The idea of this approach was first published in 1993 by Church and Hovy, as an application for "crummy" machine translation. (Church and Hovy, 1993)

We intend to design and implement a software component that is capable of aiding the translator with typing completions and can be integrated into computer assisted translation (CAT) tools.

Numerous approaches have been designed to solve this problem since the mid-1990's. The main flow of research was applying techniques elaborated for traditional statistical machine translation with some tailoring, i.e. the prediction hint were based on the machine translated results prefix filtered by the text just typed by the translator. This turned out to be rewarding, as adapting these traditional statistical techniques to the predictive scenario only required the addition of a prefix matching to the process, so the training methods could have been reused. These techniques are summarized in the Section 2.

As opposed to the above mentioned generic approaches of machine translation methods, in this paper we investigate a much simpler, language independent approach to provide efficient completion. Our method is also based on statistical processing, but instead of estimating and smoothing model parameters in iterations, it takes advantage of the large amount of memory available in state of the art computers to extract prediction information in a single step from both monolingual and bilingual corpora (training phase). Afterwards these

information are used for providing real-time completion hints for the translator (lookup phase). In this paper we investigate and compare the quality of monolingual and bilingual dictionaries created as suggested above.

The rest of the paper is organized as follows. Section 2 gives an overview of the related works. It introduces other text prediction methods and enumerates some approaches to measure and compare the quality of the provided results. In Section 3 we give an introduction to our method by defining terminology and theoretical background used in this paper, and outline the measurement framework. Section 4 describes the details of our method to capture and extract the information of existing corpora to provide prediction capabilities, and outline the evaluation method chosen to compare the different approaches. Section 5 provides measurement results about the methods in the focus of our attention. Section 6 summarizes our work and enumerates some possibilities for future improvements.

2 Related Works

Predictive typing had an important role in speeding up text input on mobile phones with hard keys (Silfverberg et al., 2000), by allowing to press a key on the phone only once for each character and disambiguating using a dictionary, i.e. trying to determine which word the user wanted to type. How and Kan (2005) suggested automatic word completion based on the keys typed and the previous word. Masui (1998) investigated the text-input productivity gain for English and Japanese language by using automatic word completion on pen-based soft key input devices, based on a predictive dictionary that stores words with their context. Approximate string matching based on spatial key layout was also presented in that work, to make the system tolerant for typos.

The groundbreaking research in predictive translation typing was the TransType project (Foster et al., 2002). Researchers of that project introduced the concept of using the production of the translation text as the key of interaction (Foster et al., 1997). That way the human translator ensures high-quality translation, while the computer can give a significant gain in productivity.

Machine translation engines designed in this project were based on IBM translation models (Brown et al., 1993). In that paper five translation models were designed by IBM. Some of them were used in TransType, and some alternative translation models were designed (e.g. maximum entropy minimum divergence model (Foster, 2000)), however, only words, or shorter sequence of words were offered as translation completion to the user.

Research were continued in the TransType2 project, a successor of TransType. Alternative translation models were designed, with the new approach to always give full sentence translation completion to the translator. In each iteration she accepts some prefix of the completion, then types some other text, and the computer gives another full sentence completion. Alignment templates, phrase-based models and stochastic finite-state transducers were examined in TransType2 as translation models. (Barrachina et al., 2009)

The performance (in the sense of "goodness") of the systems designed were assessed by comparing translations produced by the system with a reference translation of a test parallel corpus. Some of the metrics measure the quality of the translation engine without any user activity, some of them try to estimate how much work can be saved by the translator using the system. The most frequently applied goodness metrics are the following:

- **Bilingual evaluation understudy (BLEU)**: Based on how much of the reference translation text is produced ("covered") by the translation system. Usually measured in %. (Papineni et al., 2002)
- **Word error rate (WER)**: The Levenshtein-distance applied to the word strings produced by translation system and the reference translation (equals to the minimum number of substitution, insertion and deletion of words that is needed to transform the generated translation into the reference translation). It is normalized by the number of words in the reference sentences.
- **Longest common subsequence**: The number of words in the longest common subsequence of words in the generated and reference translation, divided by the number of words in the reference translation. By subsequence a sequence of words are understood, that can be created by omitting some elements of the original sequence. I.e. sub-

sequences may not be consecutive words in the original sequence.

- **Keystroke ratio (KSR)**: Ratio of number of keystrokes to the total number of characters. Measured in %.

The actual metrics used in our measurement are presented in Section 4.4.

3 Concepts and Framework

Our primary goal is to provide automatic completion for translation typing. To achieve this – similarly to the above mentioned works – we apply a dictionary based approach. We also aim to predict multiple word completions and involve information gained from the text being translated, but we chose a simpler, non-iterative way as opposed to usual statistical machine translation methods.

3.1 Terminology

For a better understanding we define here some terminology used throughout this paper.

Source/target language: The language a text is being translated from/to. E.g. From *English* to *German*.

Segment: A sentence, or group of consecutive related words (titles, bullet items, table cells) in a text, that can be translated independently from the previous and the following parts of the text. E.g. *I had to run. I was late for my bus.*

(Bilingual) parallel corpus: A large amount of segmented source language text together with its target language counterpart, where each segment in the source language text is paired (aligned) with its translation in the target language text. An example is shown in Table 1.

Subsentence: Part of a sentence that does not contain punctuation, apostrophe, quotation marks, etc. E.g. In a sentence: *I was tired, so I said "I want to go home"*, the subsentences are: '*I was tired'*; '*so I said'*; '*I want to go home'*.

Bigram/trigram: Two/three consecutive words *in a subsentence*. For the previous example: *'want to'* and *'want to go'* are considered bigrams or trigrams, but '*tired, so I'* is not.

Text unit: A word, or a set of grammatically related words in a subsentence.

3.2 Scoring

When the source language is involved in building predictive dictionaries, a fundamental problem is how to determine if a target language text unit is the translation of a source language text unit. In the example of Table 1, at first all we know that a segment is the translation of another. On a text unit level we need to determine that, e.g. the translation of "This is all in accordance" is "All dies entspricht". With statistical methods it is impossible to know for sure, hence a *score* is assigned to each (s_i; t_j) source and target language text unit pair that represents the probability of them being translation of each other.

The score is based on how often do s_i and t_j appear together in a segment and its translation, and how often one appears without the other. The former is called **concomitant occurrence**, the latter is called **supplementary occurrence**. Intuitively one would say, that if the number of concomitant occurrences are significantly greater than the number of supplementary occurrences, then t_j is the translation of s_i, otherwise it is not. A large number of concomitant occurrences is not sufficient, since e.g. the English word "the" appears almost in every sentence, so for any German word g, $count("the", g) \approx count(g)$, but "the" has only a few translation equivalents in German.

English	German
This is all in accordance with the principles that we have always upheld.	All dies entspricht den Grundsätzen, die wir stets verteidigt haben.
Thank you, Mr Segni, I shall do so gladly.	Vielen Dank, Herr Segni, das will ich gerne tun.
Indeed, it is quite in keeping with the positions this House has always adopted.	Das ist ganz im Sinne der Position, die wir als Parlament immer vertreten haben.

Table 1. English-German parallel corpus for demonstration. Source: Proceedings of the European Parliament.

Let us introduce a notation in the context of a given (s_i, t_j) pair:

- N – number of segment pairs in the input parallel corpus
- N_{11} – number of concomitant occurrences, i.e. number of segment pairs where both s_i and t_j occur
- N_{00} – number of segment pairs where neither s_i, nor t_j is found
- N_{10} – number of segment pairs where s_i occurs, but t_j does not
- N_{01} – number of segment pairs where s_i does not occur, but t_j does

Marginal counts can also be defined from the previous numbers:

- $N_{0p} = N_{00} + N_{01}$ = number of segment pairs where s_i does not occur
- $N_{1p} = N_{10} + N_{11}$ = number of segment pairs where s_i does occur
- $N_{p0} = N_{00} + N_{10}$ = number of segment pairs where t_j does not occur
- $N_{p1} = N_{01} + N_{11}$ = number of segment pairs where t_j does occur

It is obvious, that $N = N_{00} + N_{11} + N_{10} + N_{01} = N_{0p} + N_{1p} = N_{p0} + N_{p1}$.

These numbers can be arranged in a so-called *contingency table*[1]:

	$!t_j$	t_j	
$!s_i$	N_{00}	N_{01}	$\sum=N_{0p}$
s_i	N_{10}	N_{11}	$\sum=N_{1p}$
	$\sum=N_{p0}$	$\sum=N_{p1}$	$\sum=N$

The above described intuitive definition of translation relationship can be rephrased more formally with contingency tables: t_j is the translation of s_i, if N_{00} and N_{11} is dominant in the contingency table, and N_{0l} and N_{10} have low values. That means, if s_i occurs in the source language sentence, then probably t_j occurs in the target language sentence, but if s_i is missing from the source language sentence, then t_j is missing from the target language sentence.

[1] The exclamation mark (!) in the table means that the text unit after the exclamation mark does NOT appear in the corresponding segment

There are various ways to give a score based on the contingency table and this intuitive feeling (McInnes, 2004), comparing them would be subject of another study. For this experiment, we chose the *Dice coefficient comparator*:

$$score = 2 \cdot \frac{N_{11}}{N_{p1} + N_{1p}}$$

Here the score falls in the [0; 1] interval, expressing the ratio of the concomitant occurrence to the sum of individual occurrences. If it is above a threshold, which is a parameter of the algorithm, then we say *t_j is the translation of s_i*, otherwise it is not.

3.3 Framework

Before introducing our algorithms let us have a brief look on the overall workflow of evaluation the predictive performance of the algorithms. It is illustrated on Figure 1.

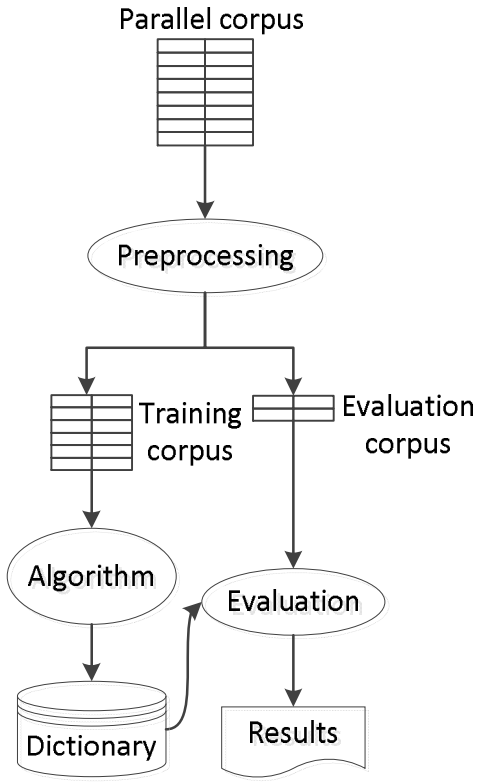

Figure 1. Evaluation process for the algorithms

The experiment is carried out on a parallel corpus. It is preprocessed and divided into two parts: *training corpus* and *evaluation corpus*. We chose the training corpus to be the size of 200k segment pairs, and the evaluation corpus to be 20k segment

pairs (10:1 ratio). The training corpus is given to the dictionary building algorithms as input, and the evaluation corpus is used to evaluate the output dictionary of the algorithms. These steps are detailed in Section 4.

This measurement process is executed multiple times with varying algorithms and/or their parameters, and the results are compared.

4 Prediction Workflow

In this section we describe the steps of our measurement framework in detail.

4.1 Preprocessing

In order to simplify and unify text handling in the algorithms, the parallel corpora need to be normalized. Performed operations include removing punctuation and braces from the end of each line and replacing subsentence separator characters (e.g. quotation mark, apostrophe, comma, colon, semicolon, etc.) with a single marker (we used |). Using this convention allows the algorithms to hinder the bi- and trigrams from crossing subsentence boundaries.

Multiple whitespaces between words are combined into a single whitespace, and all of them are replaced by a simple space.

Another important aspect of the normalization is that some special characters (related to the alphabet of the languages under inspection) needs to be expanded for the lookup and evaluation phases to work correctly. These include: 'ß' in German ('ss'), 'œ' and 'æ' in French ('oe' and 'ae' respectively) and the soft hyphen in Spanish (omitted).

An example can be seen on an English text in Table 2.

Original	Preprocessed
Please rise, then, for this minute silence.	Please rise|then|for this minute silence
(The House rose and observed a minute silence)	The House rose and observed a minute silence
Madam President, on a point of order.	Madam President|on a point of order

Table 2. Example of preprocessing. Source: Proceedings of the European Parliament.

4.2 Monolingual approach

The monolingual algorithm completely ignores the source language text, and only collects frequent text units from the target language part of the training corpus.

Frequent text units are obtained by simple counting the number of occurrences of the candidates. Multiple occurrences of a text unit within a segment count as one. Text unit with an occurrence count less than a threshold *c* are dropped in order to filter out typographical errors and hapax legomena. The threshold is defined as a percentage of the total segment count.

If *c* is set to 0 %, then every text unit is retained and saved in the predictive dictionary and no more information can be gathered from the target language file using this algorithm (danger of noise and overlearning). Text units are stored in the dictionary with their occurrence counts.

An example of the monolingual dictionary for German language can be seen in Table 3.

Text unit	Count
die	174781
der	160938
und	130495
in	71595
…	…
des Europäischen Parlaments	1698

Table 3. German monolingual dictionary in descending order of text unit appearances

When searching for completion the only thing that is used is the target language text being typed by the translator, as the building algorithm completely ignores the source text of the input corpus. For fast lookup a prefix tree is built from the dictionary when it is loaded into memory.

Completion is offered when the translator starts typing *a new word*, and completions are continuously adjusted as the user types more characters. All the items in the dictionary are matched with the prefix typed by the translator, and the best six results are given back. Six was chosen as a small number of elements that can be easily scanned by the user. Optimizing the number of displayed elements can be subject of another study.

4.3 Bilingual approach

The bilingual algorithm has basically four steps:

1. Count text unit occurrences for both source and target language, and filter out those, whose occurrence fall under a given threshold x.
2. For each remaining text unit of the source text, count the concomitant occurrences with each remaining text unit of the target text (i.e. how often they appear together in a segment and its translation).
3. For each source and target text unit pair, assign a score based on the individual and concomitant occurrences. If score falls under a given threshold s, omit this pair.
4. Add the remaining (source, target) text unit pairs to the resulting dictionary.

Step 1 is very similar to the monolingual algorithm, as it is done for the source and the target language separately. Multiple occurrences in one text segment counts as one. Filtering out rare text units helps avoid typographical errors and hapax legomena, and also helps to stay inside memory boundaries in *Step 2*.

For *Step 2*, the input parallel corpus is processed again, this time handling the source segments with their aligned translation pair together. If a source text unit s_i survived *Step 1*, and a target text unit t_j survived *Step 1*, then an occurrence counter is increased for the (s_i, t_j) pair. Multiple occurrences in a segment counts as one.

Each pair is assigned a score in *Step 3* to decide if they are translations of each other, i.e. they should be in the dictionary.

In *Step 4* a dictionary is built from the source language text units and its translations. An entry in the dictionary contains the source language text unit and a list of possible translations with the assigned scores. An example can be seen in Table 4.

Source language text unit	Possible translations
I would	(möchte; 0.45), (möchte ich; 0.31)
new	(neuen; 0.55), (neue; 0.49)
Parliament	(Parlament; 0.75), (Parlaments, 0.40)
and I	(und ich; 0.54)
and	(und; 0.88), (der; 0.61), (die; 0.59), (in; 0.46), …

Table 4. Bilingual dictionary fragment for English-German language pair

When looking for translation completions, the source language segment is split into text units and those text units are looked up in the dictionary. This way the dictionary gives target language text units that possibly appear in the translation, as they have counterparts in the source language segment. These are filtered by the prefix of the next word the translator has typed, ordered by the score, and the best six is shown to the user.

4.4 Evaluation

Last step of the framework is measuring the goodness of the algorithms.

The input of the evaluation process is the evaluation corpus and the predictive dictionary produced by the algorithms. We seek to measure how much help the dictionary would provide, if the translator was typing the translation found in the evaluation corpus.

The process calculates two metrics for each segment pair of the evaluation corpus, and the output is the average of these metrics. We chose two metrics to see if they correlate in this case, as they do when applied to statistical machine translation systems. The two metrics are:

- **Coverage**: Measures how much of the translation can be found in the dictionary. Expressed in %. Basically the same as BLEU.
- **Keystroke saving ratio**: The ratio of the keystrokes saved by the predictive translation typing to the total number of characters. Also measured in %, and closely related to keystroke ratio.

As mentioned in Section 2, coverage measures the quality of the dictionary by adding up the length of all the hits in the target text, while keystroke saving ratio models the effort saved by the translator using the system. To be precise, as multiple suggestions are shown to the translator, if she chooses e.g. the 5th element in the list, then the keystroke saving is decreased by 4, modelling that she had to press the down arrow four times to choose that completion.

Also, to be fair, the comparison is based on the actual size (e.g. in megabytes) of the dictionary, that is we seek to compare the quality of equally sized dictionaries produced by the different algorithms (the actual size of dictionaries is controlled

by the threshold parameters we apply during extraction).

5 Measurement Results

Measurements were carried out for English-German, English-French and English-Spanish language pairs. The parallel corpora were extracted from the proceedings of the European Parliament, downloaded from the OPUS project (Tiedemann, 2012).

An aspect of the measurement was the exact interpretation of the *text unit* concept. Here two kinds of text unit was investigated for each algorithm: one just allows *words*, the other includes *words*, *bigrams* and *trigrams* (as defined in Section 3.1).

Table 5 summarizes the dictionary building times for each case, when the size of the output dictionary was around 1 MB / 10 MB. It clearly shows that creating a bilingual dictionary requires more time than creating a monolingual. However, it is a one-time investment (as usually dictionaries are created once and used often), and it is refunded by the greater keystroke saving capabilities of the bilingual dictionary.

Measurement results can be seen on Figure 2-Figure 4. As mentioned in the previous section, the comparison is based on the size of the dictionary, so the dictionary size is on the horizontal axis (with logarithmic scale!), and the average coverage / keystroke saving ratio is displayed on the vertical axis. On the legend, 'ml'/'bl' refers to monolingual/bilingual, and 'w'/'wbt' refers to that bigrams and trigrams were not/were included.

It can be concluded from the results that with greater dictionary size comes greater coverage and keystroke saving. The slope of the curves is similar for the three inspected language pairs.

Using bi- and trigrams (dotted vs solid lines) requires larger dictionary sizes for the same coverage and keystroke saving values, especially in the 100 kB – 3 MB interval. Over 3 MB similar coverage plateau values can be reached (95-97 % for monolingual and 70-80 % for bilingual), but keystroke saving values are greater with bi- and trigrams, especially for the bilingual algorithm. However, this gain is little, only about 3-5 %.

In the aspect of monolingual and bilingual approach, the monolingual algorithm produces great coverage values and reaches its plateau at small dictionary sizes (1-10 MB). The top of its keystroke saving ratio is around 28 %, which can be obtained with a 10 MB dictionary. A monolingual dictionary of this size can be created within 17 seconds. The bilingual algorithm always reaches less coverage, but it is better at keystroke saving. The top of keystroke saving (38 %) is 6-12 % better than the top keystroke saving of the monolingual algorithms.

A little exception can be observed here with the English-German language pair. The top of monolingual keystroke saving values is 3-4 % greater than the two other language pairs, and for small dictionary sizes the keystroke saving values are greater than that of the bilingual algorithm.

Finally, coverage and keystroke saving values does not correlate in this case: the monolingual algorithm is better at coverage, but the bilingual algorithm has better keystroke saving values. Since keystroke saving measures the real effort saved by the translator, coverage should not be used for comparison, even if it could be more easily measured than keystroke saving. Hence the bilingual algorithm can be considered better.

Language pair	**Training corpus size** [kB]	**Monolingual, words only** [mm:ss]	**Monolingual, words, bi- and trigrams** [mm:ss]	**Bilingual, words only** [mm:ss]	**Bilingual, words, bi- and trigrams** [mm:ss]
English-German	62 146	00:03 / -	00:25 / 00:17	01:21 / 01:21	05:39 / 05:09
English-French	62 196	00:03 / -	00:27 / 00:17	01:32 / 01:30	06:46 / 06:24
English-Spanish	61 019	00:03 / -	00:27 / 00:18	01:28 / 01:28	06:22 / 06:02

Table 5. Corpus sizes and dictionary creation times for dictionary size of 1 MB / 10 MB. Dashes in the first monolingual column means that maximum dictionary size in that case was less than 10 MB.

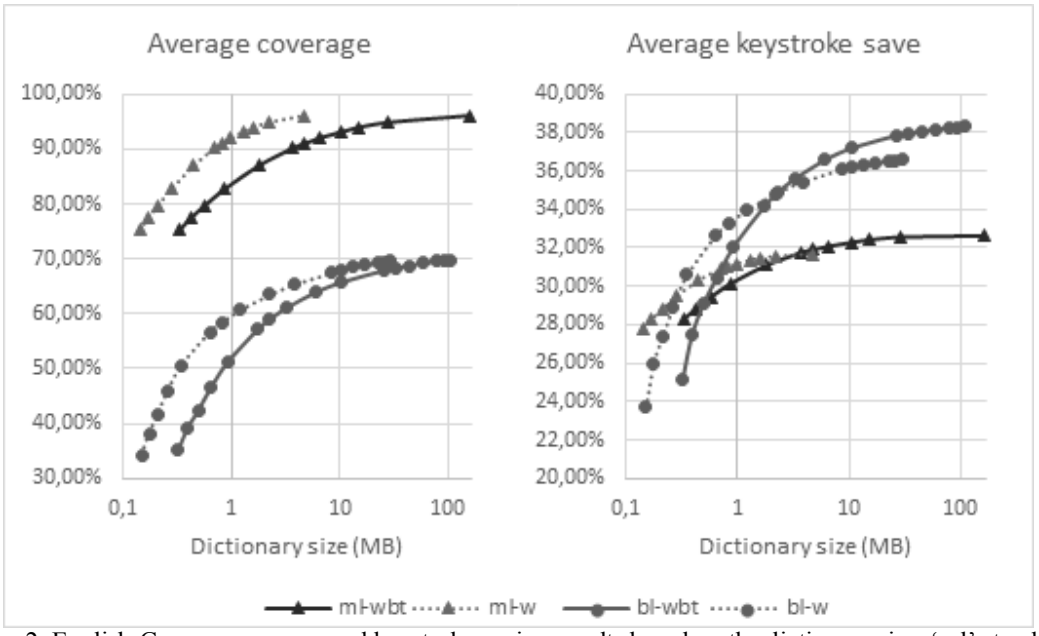

Figure 2. English-German coverage and keystroke saving results based on the dictionary size. 'ml' stands for monolingual; 'bl' stands for bilingual; 'w' stands for only words, no bigrams or trigrams; 'wbt' stands for words with bigrams and trigrams.

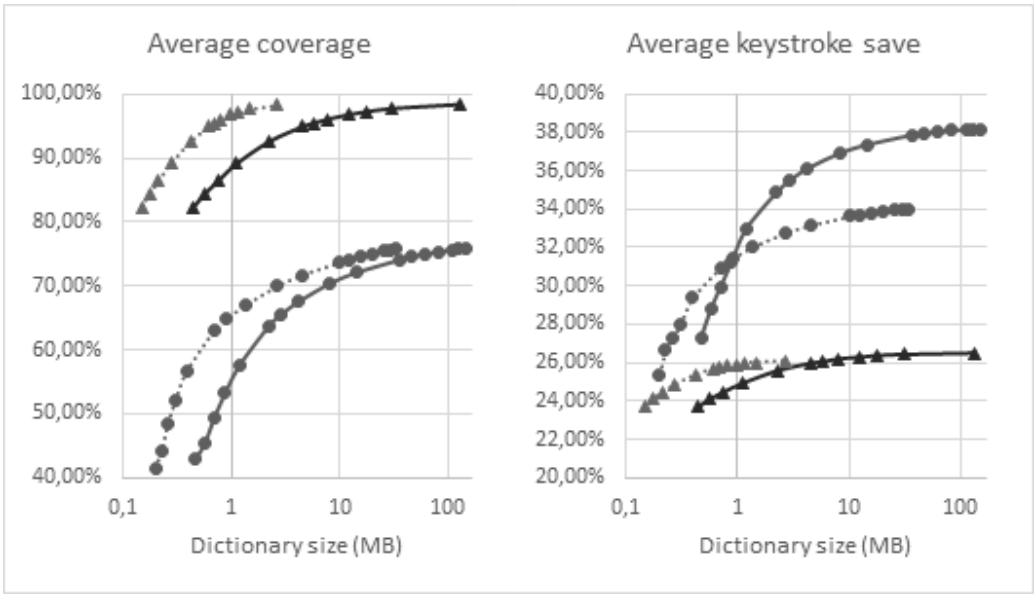

Figure 3. English-French coverage and keystroke saving results based on the dictionary size. Legend is the same as on Figure 2.

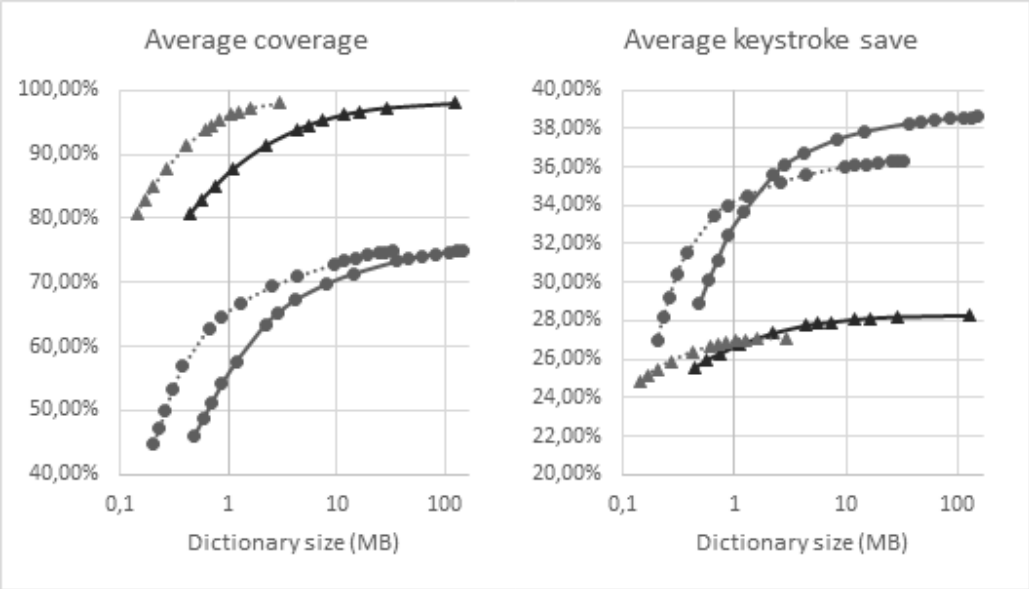

Figure 4. English-Spanish coverage and keystroke saving results based on the dictionary size. Legend is the same as on Figure 2.

6 Summary

This paper investigated the effect of using bigrams and trigrams, and the effect of using the source language in a dictionary for predictive translation typing. Using bigrams and trigrams gave little gain (3-5 %), but using the source language resulted in a more significant improvement, about 10-12 % of the keystrokes can be additionally spared compared to the monolingual frequency counting approach.

Our approach is simpler than those statistical machine translation based ones used in TransType and TransType2 projects. In our method, instead of learning, smoothing and optimizing translation model parameters, normal and concomitant occurrences are counted directly and given a score based on the result.

As an alternative approach, the algorithms also can be tweaked in numerous ways. For monolingual predictions the previous *n* words could also be taken into account when showing the probable completions. For the bilingual algorithm preferring text unit pairs that are similar length could improve performance, because translation of a short text tends to be short. Another improvement could come from using another statistical probing method for calculating score for text unit pairs, or making text units of arbitrary words inside subsentences, not just from consecutive ones.

References

Allen, Jeffrey. "Post-editing." *Benjamins Translation Library* 35 (2003): 297-318.

Barrachina, Sergio, et al. "Statistical approaches to computer-assisted translation." *Computational Linguistics* 35.1 (2009): 3-28.

Brown, Ralf D., and Sergei Nirenburg. "Human-computer interaction for semantic disambiguation." *Proceedings of the 13th conference on Computational linguistics-Volume 3*. Association for Computational Linguistics, 1990.

Brown, Peter F., et al. "The mathematics of statistical machine translation: Parameter estimation." *Computational linguistics* 19.2 (1993): 263-311.

Church, Kenneth W., and Eduard H. Hovy. "Good applications for crummy machine translation." *Machine Translation* 8.4 (1993): 239-258.

Foster, George. "A maximum entropy/minimum divergence translation model."*Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2000.

Foster, George, Pierre Isabelle, and Pierre Plamondon. "Target-text mediated interactive machine translation." *Machine Translation* 12.1-2 (1997): 175-194.

Foster, George, Philippe Langlais, and Guy Lapalme. "TransType: text prediction for translators." *Proceedings of the second international con-*

ference on Human Language Technology Research. Morgan Kaufmann Publishers Inc., 2002.

How, Yijue, and Min-Yen Kan. "Optimizing predictive text entry for short message service on mobile phones." *Proceedings of HCII*. Vol. 5. 2005.

Masui, Toshiyuki. "An efficient text input method for pen-based computers."*Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co., 1998.

McInnes, Bridget T. *Extending the log likelihood measure to improve collocation identification*. Diss. UNIVERSITY OF MINNESOTA, 2004.

Papineni, Kishore, et al. "BLEU: a method for automatic evaluation of machine translation." *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002.

Silfverberg, Miika, I. Scott MacKenzie, and Panu Korhonen. "Predicting text entry speed on mobile phones." *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 2000.

Tiedemann, Jörg. "Parallel Data, Tools and Interfaces in OPUS." *LREC*. 2012.