Input Method for Human Translators: A Novel Approach to Integrate Machine Translation Effectively and Imperceptibly

GUOPING HUANG, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, University of Chinese Academy of Sciences, Tencent AI Lab
JIAJUN ZHANG and YU ZHOU, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, University of Chinese Academy of Sciences
CHENGQING ZONG, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, University of Chinese Academy of Sciences, CAS Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences

Computer-aided translation (CAT) systems are the most popular tool for helping human translators efficiently perform language translation. To further improve the translation efficiency, there is an increasing interest in applying machine translation (MT) technology to upgrade CAT. To thoroughly integrate MT into CAT systems, in this article, we propose a novel approach: a new input method that makes full use of the knowledge adopted by MT systems, such as translation rules, decoding hypotheses, and n-best translation lists. The proposed input method contains two parts: a phrase generation model, allowing human translators to type target sentences quickly, and an n-gram prediction model, helping users choose perfect MT fragments smoothly. In addition, to tune the underlying MT system to generate the input method preferable results, we design a new evaluation metric for the MT system. The proposed input method integrates MT effectively and imperceptibly, and it is particularly suitable for many target languages with complex characters, such as Chinese and Japanese. The extensive experiments demonstrate that our method saves more than 23% in time and over 42% in keystrokes, and it also improves the translation quality by more than 5 absolute BLEU scores compared with the strong baseline, i.e., post-editing using Google Pinyin.

CCS Concepts: • **Computing methodologies** → **Machine translation**; • **Human-centered computing** → **Text input**;

Additional Key Words and Phrases: Machine translation, computer-aided translation, input method, evaluation metric

ACM Reference format:
Guoping Huang, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2018. Input Method for Human Translators: A Novel Approach to Integrate Machine Translation Effectively and Imperceptibly. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 18, 1, Article 4 (November 2018), 22 pages.
https://doi.org/10.1145/3230638

The bulk of the research was done at CASIA. The research work descried in this article has been supported by the National Key Research and Development Program of China under Grant No. 2016QY02D0303 and the Natural Science Foundation of China under Grant No. 61333018.
Authors' addresses: G. Huang, J. Zhang, Y. Zhou, and C. Zong (corresponding author), 7-th Floor, Intelligence Building, No. 95, Zhongguancun East Road, Haidian District, Beijing, 100190, China; G. Huang is currently affiliated with Tencent AI Lab, Netac Building, High-tech Sixth South Road, Nanshan District, Shenzhen; emails: {guoping.huang, jjzhang, yzhou, cqzong}@nlpr.ia.ac.cn; G. Huang's current email is donkeyhuang@tencent.com.

https://doi.org/10.1145/3230638

1 INTRODUCTION

Computer-aided translation (CAT) is a form of language translation in which a human translator uses software to perform and facilitate the translation process. To further improve the translation efficiency, incorporating machine translation (MT) technology, particularly statistical machine translation (SMT), into CAT tools has drawn increasingly more attention. Researchers have proposed many approaches, which can be divided into two types. The first aims to develop interactive machine translation (IMT) systems (Foster and Lapalme 2002; Barrachina et al. 2009; Cheng et al. 2016; Huang et al. 2016a). The second focuses on designing good post-editing (PE) systems (Koehn 2009a, 2009b; Carl et al. 2011; Koehn 2012; Zhechev 2012; Koehn et al. 2014).

In IMT, the core idea is to help an MT system dynamically generate an acceptable translation in a left-to-right manner through a series of timely interactions between human translators and the MT system. Human translators are required to observe and carefully revise the MT output. To the best of our knowledge, IMT is not widely accepted or adopted by any commercial CAT tools because of the heavy workload of the translation process.

In practice, post-editing is a standard and widely used approach to apply MT technology to upgrade CAT systems: Human translators accomplish the translation work by modifying the MT outputs. If the raw MT output is good enough, then it will take little time for human translators to achieve the final acceptable translation. A considerable amount of evidence has shown that human translators are more productive and that the translation results are more accurate when post-editing is adopted (Carl et al. 2011; Koehn 2012; Zhechev 2012). In industry, there are a number of CAT tools that support post-editing, such as SDL Trados and MemoQ.

However, post-editing is far from perfect. There are two main challenges for post-editing in practice. First, at the present stage, the low quality of MT results often makes a human translator unwilling to edit, and he/she would rather ignore the MT results and start translating from scratch. Second, many target languages (e.g., Chinese and Japanese) are written in complex character sets. To type these complex characters into a computer, human translators have to use a specially developed input method, such as Google Pinyin, which allows users to input Chinese characters by entering phonetic spellings. In Chinese, a phonetic spelling generally matches dozens of Chinese characters, and, thus, it leads to many more editing operations.

Fortunately, even if the final MT result is terrible and is ignored by human translators in post-editing, it may contain some good fragments. Therefore, it raises the question of how to take advantage of such fragments. Moreover, all human translators that translate other language texts into Chinese (or other languages with complex characters) need an input method. This fact inspires us to consider integrating MT technology into the input method to accelerate the human translation process. Why not avoid the intensive interaction between human translators and the low-quality MT outputs by imperceptibly applying only the reliable information into the input method?

To achieve these goals, as shown in Huang et al. (2015), we propose a novel approach that thoroughly integrates MT into the CAT system: a well-designed input method named CoCat, which makes full use of the knowledge adopted by MT systems, such as translation rules, decoding hypotheses, and n-best translation lists. First, we analyze and extract the useful information of the underlying MT system, and we transform such information into features. Second, we extend the standard input method and design a log-linear model to incorporate multiple sources of features, including extracted MT-related features. Third, we design an n-gram prediction model for the input method to further facilitate the human translators.

Moreover, to tune the underlying MT system to generate input method preferable results, as shown in Huang et al. (2016c), we design a novel MT evaluation metric, MinKSR, for the MT system in this article. The MinKSR evaluation metric makes longer perfect fragments, corresponding to fewer keystrokes. The word "minimum" means that the value of MinKSR shows how many

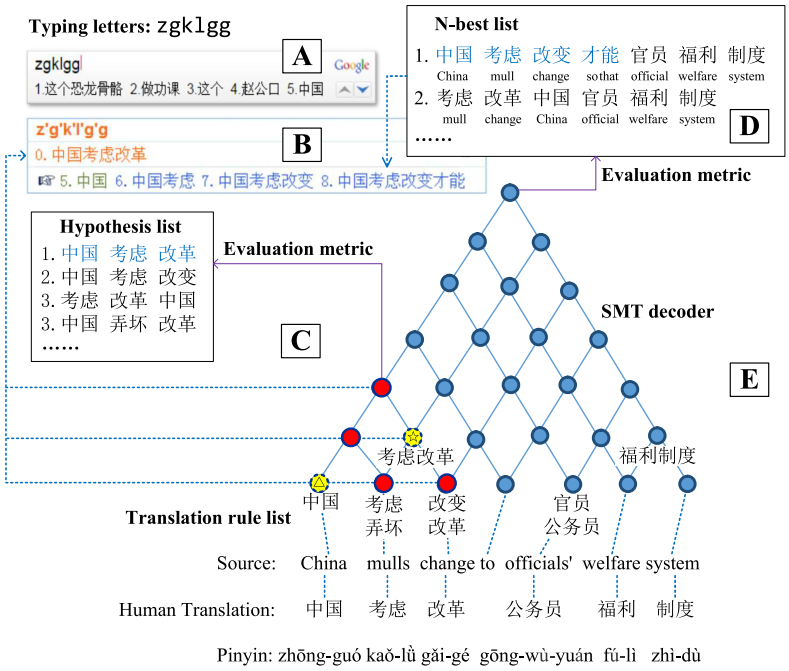


Fig. 1. An overview of the CoCat input method. *Note*: Each circular node refers to a decoding span. The Chinese phrases below the nodes refer to the phrase translation rules and hypotheses during translation decoding.

keystrokes can be saved at least by the MT results. The higher the value of MinKSR is, the more keystrokes that can be saved.

Our proposed CoCat input method works well in both scenarios: translating from scratch and post-editing. Figure 1 demonstrates how the approach works in translation from English to Chinese. If the human translator adopts the standard Google Pinyin to perform translation from scratch, then the abbreviated Chinese typing letters "zgklgg" (the acronym Chinese Pinyin) cannot elicit the correct translation (in Zone A) because Google Pinyin cannot perceive what exactly the current user is translating. Rather, in Zone B, our proposed CoCat input method can correctly decode the abbreviated letters "zgklgg" into the desired result "中国考虑改革(China considers to reform)" with the help of translation rules and hypotheses in Zone C, all of which are used by the underlying MT system (in Zone E). Meanwhile, the CoCat input method provides an n-gram prediction list (5-based ordinal in Zone B) based on the n-best list (in Zone D) provided by the MT system. Furthermore, under the guidance of the MinKSR evaluation metric, the SMT decoder generates more suitable deep information required by the CAT-oriented input method (in Zones C and D). In this way, the CAT-oriented input method greatly improves the productivity of human translation.

In the experiments, our proposed input method has achieved remarkable results on professional human translation tests. This article provides the following contributions:

(1) The proposed CoCat input method takes full advantage of useful information of the MT system. CoCat is the first input method to exploit the deep information used by MT, such as translation rules and decoding hypotheses.

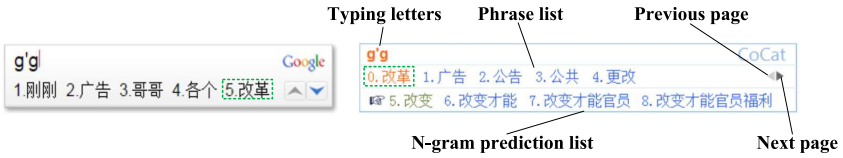


Fig. 2. The comparison between the Google Pinyin and the CoCat input method.

(2) CoCat helps human translators significantly save time and keystrokes, as well as substantially improves the final translation quality.
(3) CoCat can collaborate with other CAT technologies, for example, post-editing. When CoCat is integrated with post-editing, it can further accelerate the translation process.
(4) Under the guidance of the MinKSR evaluation metric, the underlying MT system generates more useful deep information for coordinating human translators with the CAT-oriented input method. It helps the input method substantially and firmly reduce the keystrokes of the translating process.

2 COCAT INPUT METHOD

In this article, we focus on English-to-Chinese translation, which is a typical case for translating texts into languages with non-alphabetic characters and that requires a specific input method. After applying MT to CAT tools, the MT result will be evaluated and used in three ways according to its quality: (1) the MT output is perfect, and it happens to be the one that the human translator expects; (2) the MT output is good but not perfect, and the human translator needs to perform some minor modifications with post-editing; and (3) the MT output is of poor quality, and the human translator will simply ignore it.

Generally, good MT results greatly facilitate human translators when using post-editing. However, in most cases, the final MT result is not good enough. There are only a few perfect fragments, and tedious modifications are required.

Based on the above analysis, we propose an input method called CoCat that provides a novel interactive approach to integrate MT by making full use of those perfect fragments. The user interface of CoCat is shown in Figure 2. Similarly to Google Pinyin, the CoCat user interface contains typing letters and the corresponding phrase list. The previous page button and the next page button are used to switch between phrase pages. Compared with Google Pinyin, the extra part of the CoCat input method is the n-gram prediction list, which is similar to the auto suggestions or word associations of the other Chinese input methods.

The primary difference from the other input methods is that the CoCat input method integrates the MT knowledge into the typing process in the translation scenario. Thus, the CoCat input method includes two novel CAT-oriented models: the phrase generation model and the n-gram prediction model. Figure 1 and Figure 2 illustrate the superiority of the two models compared with traditional input methods in three ways:

(1) The phrase generation model re-ranks the target phrase list in the current context with additional features induced from the MT system. For example, “改革(reform)” is ranked first by CoCat among all the candidates for the typing letters “gg,” whereas Google Pinyin will not provide the desired result, as shown in Figure 2.
(2) The phrase generation model produces new target phrases for the current source sentence with the aid of deep information embedded in the underlying MT system. For example, typing the letters “zgklgg” using Google Pinyin cannot lead to the expected phrase

"中国考虑改革(China mulls to change)." However, typing the same letters can achieve the expected result using the CoCat input method, as shown in Figure 1.

(3) The n-gram prediction model offers a list of translation suggestions. For example, it provides four potential phrases from indices 5 "改变(change)" to 8 "改变才能官员福利," as shown in Figure 2.

2.1 Phrase Generation Model

On the one hand, a phonetic spelling generally matches dozens of characters in Chinese. Therefore, it is very difficult for input methods to automatically provide the correct Chinese phrase. The user types phonetic spellings, which are called *typing letters* in this article, to input a Chinese phrase. For example, to input the target phrase "中国考虑改革(China mulls to change)" using an input method, such as Google Pinyin, the translator typically types the Chinese pinyin with six phonetic spellings "zhongguokaolvgaige." The input method automatically segments the typing letters into "zhong'guo'kao'lv'gai'ge" and finally converts it into a large candidate set of Chinese phrases, including "中国考虑改革" whose scores may rank low among other candidates. To accelerate the typing process, the CoCat input method will make the expected phrase rank as high as possible according to the current context.

On the other hand, we attempt to greatly reduce the number of keystrokes to speed up typing. The smaller the number of keystrokes is, the faster the translating is, the more time that is left for human translators to think, and the better the translation is. If we type the shorter phonetic spelling "zgklgg," then the correct result "中国考虑改革" is expected to still rank first according to the context in Figure 1. However, all the existing Pinyin input methods, including Google Pinyin, fail to produce the correct result and cannot correctly decode "zgklgg." To achieve the goal, the CoCat input method will produce a new target phrase for the current source sentence with the aid of deep information embedded in the underlying MT system.

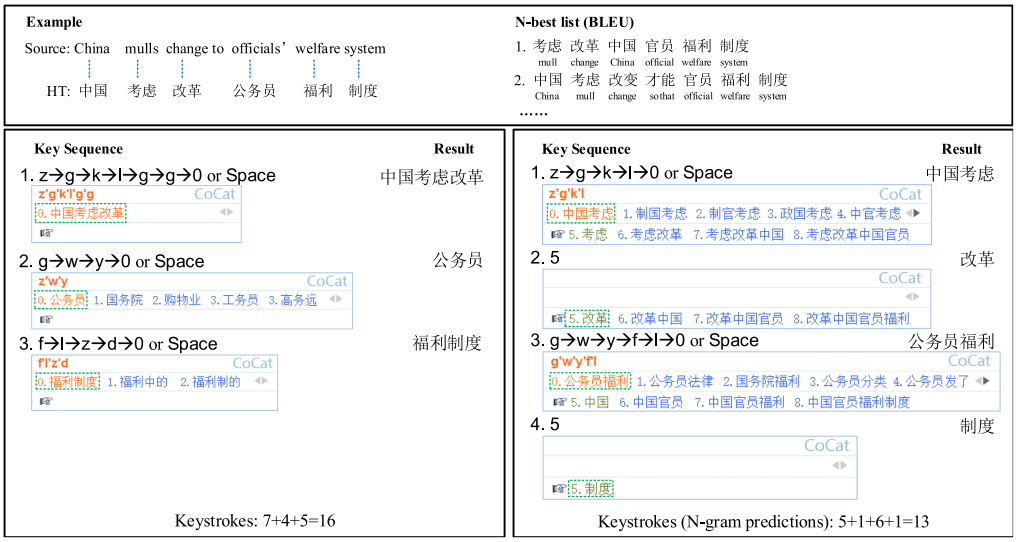
For a given segmented pinyin $y_1^n = y_1 y_2 \dots y_n$, the goal of the conversion from the pinyin to the string of Chinese characters is to find the most probable phrase $h_1^n = h_1 h_2 \dots h_n$ from the Chinese character candidate set, H, by maximizing $Pr(h_1^n | y_1^n)$. In general, y and h have the same length n, y_i is a syllable of a Chinese character, and h_i is one of the characters that y_i responds to. To better integrate the MT system, we design a new phrase generation model for CoCat based on the log-linear model:

$$\widehat{H}\left(y_1^n\right) = \underset{H}{\operatorname{argmax}} \sum_{m=1}^{M} \lambda_m f_m\left(h_1^n, y_1^n\right), \tag{1}$$

where λ_m denotes the weight of the corresponding feature and M refers to the number of feature functions. The feature function set, $\{f(h_1^n, y_1^n)\}$, includes typical features employed by traditional input methods, such as the word frequency and log-probabilities for the typing model $Pr(y_1^n | h_1^n)$ and the language model $Pr(h_1^n)$. In addition, the following three features, which are induced by the MT systems, are employed by the log-linear model:

(1) the feature function indicating whether the candidate is included in the translation rules corresponding to the current source sentence,
(2) the feature function indicating whether the candidate is included in the hypotheses during MT decoding,
(3) the feature function indicating whether the candidate is included in the n-best list of the MT output.

Decoding is performed by the CYK algorithm (Kasami 1965; Younger 1967), and a beam-search algorithm is employed to accelerate decoding.



(a) The key sequence without the n-gram prediction model (b) The key sequence with the n-gram prediction model

Fig. 3. The comparison of key sequences without/with the n-gram prediction model.

For example, to complete the translation task using the CoCat input method, we need to type the key sequence as shown in Figure 3(a). We can obtain the correct result by typing the letters "zgklgg" when we translate the sentence in Figure 1. This is because the candidate substring "中国(China)" is included in the MT translation rules (△ in Figure 1) and another candidate substring "考虑改革(mulls to change)" is contained in the MT translation hypotheses (☆ in Figure 1). The expected candidate "中国考虑改革(China mulls to change)" will be rewarded during pruning and re-ranking as shown in Step 1 in Figure 3(a). Clearly, we can achieve the same effect on other languages in the same way.

2.2 N-Gram Prediction Model

To further reduce the number of keystrokes during the entire human translation process, we propose an n-gram prediction model for the CoCat input method.

Given an MT n-best list $C = \{c_i | 0 < i \leq |C|\}$, the ith machine translation candidate is $c_i = c_{i1}c_{i2}\ldots c_{i|c_i|}$, where $|C|$ denotes the size of the n-best list, $|c_i|$ denotes the word number of c_i, and c_{ij} denotes the jth word in c_i. Suppose that the desired human translation is $t_1^m = t_1 t_2 \ldots t_m$. We let W denote the number of predictions.

In this article, a shorter n-gram prediction is actually a prefix of any longer prediction. The first prediction has only one word. The second prediction has two words, and the first word is that of the first prediction. For example, {"中国(China)," "中国 官员(Chinese officials)," "中国 官员 福利(Chinese offcials' welfare)," "中国 官员 福利 制度(Chinese officials' welfare system)"} can be a correct prediction list, while {"中国(China)," "官员(officials)," "官员 福利(officials' welfare)," "中国 官员 福利 制度(Chinese officials' welfare system)"} is not allowed. As a result, W also refers to the word number of the longest n-gram prediction. The algorithm for generating n-gram predictions is given as Algorithm 1.

According to Step (1) in Algorithm 1, the n-gram prediction model first generates initial n-gram predictions $\{c_{11}, c_{11}c_{12}, c_{11}c_{12}c_{13}, \ldots, c_{11}c_{12}\ldots c_{1W}\}$ based on the best MT candidate c_1 before the translator starts to translate. At this point, the human translation prefix does not exist, and the

ALGORITHM 1: Generating N-gram Predictions

Input: MT n-best list $C = \{c_i | 0 < i \leq |C|\}$, and human translation prefix $t_1^j = t_1\, t_2 \ldots t_j$.
Output: N-gram prediction list $P = \{p_l | 0 < l \leq W\}$.
# (1) Generate initial n-gram predictions based on the best MT candidate c_1 before typing.
if $t_1^j = NULL$ **then**
    $P = \{p_l | p_l = c_{11} \ldots c_{1l} (1 \leq l \leq W)\}$; return;
end
# (2) Generate n-gram predictions based on the human translation prefix and the MT n-best list when once the translator has typed the word t_j of the human translation t_1^m.
repeat
    $Found = FALSE$;
    $SufEnd = j$;
    # Find n-gram predictions using the longest suffix matching algorithm.
    for $SufStart \leftarrow 1$ *to* j **do**
        $Suffix = t_{SufStart} \ldots t_{SufEnd}$;
        for $i \leftarrow 1$ *to* $|C|$ **do**
            if *Suffix is found in* c_i *and the match point is* $c_{ik} = t_{SufEnd}$ **then**
                $P = \{p_l | p_l = c_{i(k+1)} \ldots c_{i(k+l)} (1 \leq l \leq W)\}$;
                $Found = TRUE$;
                break;
            end
        end
        if *Found* **then**
            break;
        end
    end
    if *Not Found* **then**
        $P = \{\}$;
    end
    # The translator can press a certain numeric key to select a corresponding prediction or ignore all the predictions and just continue typing his own translation.
until *the translator completes the entire translation process*;

longest prediction is the first W words of the MT candidate c_1. The translator can press a certain numeric key to select a corresponding prediction or ignore all the predictios and just continue typing his own translation.

According to Step (2) in Algorithm 1, the n-gram prediction model will generate n-gram predicitons based on the human translation prefix and the MT n-best list later in the translation process. The core idea of Step (2) is to find n-gram predictions using the longest suffix matching algorithm. We first employ the whole human translation prefix t_1^j to match all machine translation candidates from the best c_1 to the last $c_{|}C|$. If it succeeds, then the match point is $c_i k$, where k means that the kth word of the machine translation candidate c_i matches the last word t_j of t_1^j. And the n-gram prediction list consists of words following the match point: $P = \{p_l | p_l = c_{i(k+1)} \ldots c_{i(k+l)} (1 \leq l \leq W)\}$. If it fails, then we then use a shorter suffix t_2^j of the human translation to find a match point, and so on. If the shortest suffix t_j^j still fails to match any machine translation candidate, then the n-gram prediction model output the empty result $P = \{\}$.

For the number of predictions, a large W may save more keystrokes, but too many predictions will impose an additional burden on translators' decision making and selecting. In this article, we choose $W = 4$ as the default value.

For example, if we enable the n-gram prediction model, then the key sequence would be further optimized as shown in Figure 3(b). In Step 2, the correct prediction "改革(change)" is produced when we start translating, and we can press the numeric key 5 to choose it directly. Additionally, in Step 4, the n-gram prediction model generates the correct prediction "制度(system)," and we can press the numeric key 5 to select the suitable prediction. As shown in Figure 3, using the n-gram prediction model, we can save the keystrokes by:

$$\frac{16-13}{16} \times 100\% = 18.75\%.$$

In this way, the CoCat input method provides effective interactions and reduces the keystrokes to the greatest extent possible even if the MT results are not good enough.

In summary, the proposed CoCat input method will improve the human translation process in two aspects: (1) human translators do not need to evaluate the MT output, and (2) the new input method automatically accelerates the translation process by saving keystrokes and providing suggestions in a friendly way.

3 INPUT METHOD ORIENTED EVALUATION METRIC

In Figure 1, the SMT decoder scores translation rules, translation hypotheses and n-best candidates using a log-linear model:

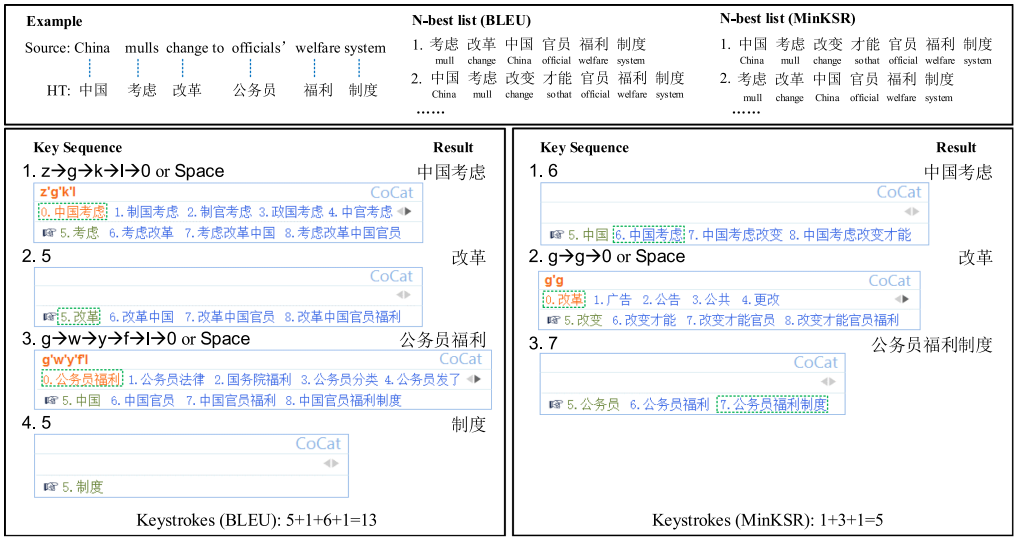
$$\log p(t|s) = \sum_{m=1}^{M} \lambda'_m f_m(t,s), \tag{2}$$

where s and t denote the source sentence and the translation candidate, respectively. In general, the features, $\{f(t,s)\}$, of the log-linear model are the probabilities from the language model, translation model, and reordering model, plus other features. Tuning is the process of finding the optimal weights, $\{\lambda'\}$, for this linear model. In the SMT tuning process, the performance of the MT system is typically measured with a certain evaluation metric that compares the final MT results with the specified references. Then, during SMT decoding, the optimal weights directly influence the deep information required by the CoCat input method, such as translation rules, translation hypotheses and the n-best list.

However, the existing MT evaluation metrics, including BLEU, TER, KSR, and KSMR, failed to cover such deep information required by the CoCat input method. The well-known corpus-level metric BLEU is based on the n-gram matching between the final MT output and the reference translations. Another popular metric, TER, measures the amount of editing needed to modify the MT output to exactly match a reference translation, and it works well in the post-editing scenario (Carl et al. 2011; Koehn 2012; Zhechev 2012). KSR (keystroke ratio) and KSMR (keystroke ratio plus mouse-action ratio) are used to estimate the effort needed, particularly in the IMT scenario (Foster and Lapalme 2002), to produce correct translations. Moreover, KSMR and KSR do not take languages with complex characters for which an input method is required, such as Chinese and Japanese, into account.

Furthermore, in the CAT scenario with the input method, we find that translators prefer to directly select the correct n-gram predictions. Consequently, it is important to generate the perfect beginnings and longer matched fragments for the CAT-oriented input method, even if the scores of BLEU or TER do not really change.

To achieve our goals, in this article, we propose a novel MT evaluation metric, MinKSR, which considers deep information required by the CoCat input method, including n-best candidates,



(a) The key sequence based on BLEU (b) The key sequence based on MinKSR

Fig. 4. The comparison of key sequences based on the MT tuned by BLEU and MinKSR.

hypotheses, and translation rules. It makes longer perfect fragments embedded in deep information correspond to fewer keystrokes.

To illustrate how MinKSR works in the CAT scenario, let us consider the example in Figure 4. The human translation refers to the target sentence in mind. For simplicity, deep information of the SMT decoder and the full n-best candidates are omitted. As we can see, 13 keystrokes are required to input the human translation with the MT system tuned by BLEU, as shown in Figure 4(a).

In contrast, in Figure 4(b), the number of keystrokes will be reduced to 5 by using MinKSR to tune the underlying MT system. The key is Steps 1 and 3. Step 1 generates the ideal prediction "中国考虑" by grasping the perfect beginning fragment. Step 3 hits the perfect prediction ("公务员福利制度," the welfare system of civil service) by searching the optimized MT deep information. This indicates that the deep information is very important for coordinating human translators with the CAT-oriented input method.

3.1 MinKSR

The purpose of the MinKSR metric is to measure how many keystrokes can be reduced at least by the MT system integrated in the MT-based input method. The core idea of MinKSR is to map the longer perfect fragments to fewer keystrokes. To automatically evaluate the MT system, MinKSR is calculated by the empirical keystrokes rather than the practical keystrokes. There are three sufficient statistics to calculate MinKSR:

(1) $mk_{norm}(t_1^m)$: the count of minimum keystrokes to input the human translation $t_1^m = t_1 t_2 \dots t_m$ using the general input method character by character without the aid of the MT system.
(2) $ek(Q, t_1^m)$: the count of empirical keystrokes to input the human translation t_1^m using the MT-based input method with the aid of the intermediate and final MT result Q.
(3) $pk(t_1^m)$: the count of minimum ideal keystrokes to input the human translation t_1^m using the MT input method with the aid of the perfect MT candidate $c_1^n = t_1^m$.

Let $s_1^j = s_1 s_2 \dots s_j$ denote the source sentence, C denote the n-best list $\{c\}_1^{|C|}$, H denote the hypothesis lists $\{h\}_1^{|H|}$, and L denote the translation rule lists $\{l\}_1^{|L|}$, where j refers to the word number of the source sentence, $|C|$ refers to the length limitation of the n-best list, $|H|$ refers to the length limitation of the hypothesis list for each phrase, and $|L|$ refers to the length limitation of the translation rule list for each phrase. Then, the MT intermediate and final results can be defined as the triple $Q = (C, H, L)$, and the number of phrases is $\frac{j\times(j+1)}{2}$ for phrase-based SMT systems. Given the reference translation $t_1^m = t_1 t_2 \dots t_m$ of the source sentence, MinKSR, r, is given as follows:

$$r(Q, t_1^m) = \frac{mk_{norm}(t_1^m) - ek(Q, t_1^m)}{mk_{norm}(t_1^m) - pk(t_1^m)}. \tag{3}$$

If there is only the candidate $c_1^n = c_1 c_2 \dots c_n$ in the MT result without deep information, then MinKSR degenerates into a general MT evaluation metric and performs similarly to BLEU. If there is more than one reference, then we simply select the minimum $ek(Q, t_1^m)$. To calculate the three terms in Equation (3), we introduce the following notations:

(1) sn: the character number of separators between words. $sn = 0$ for Chinese, and $sn = 1$ for English (the space between words).
(2) kc: the count of keystrokes to select a certain candidate or prediction from the input method. In general, $kc = 1$ without page turning.
(3) rk: a ratio dividing the character number of a word by the number of keystrokes with the help of an input method. The value of rk varies from language to language. For Chinese, in this article, we simply choose $rk = 2$ to guarantee that MinKSR is indeed a lower bound.

Then, we can count the three sufficient statistics in Equation (3) as follows:

(1) The count of minimum keystrokes using a general input method without the MT system:

$$mk_{norm}\left(t_1^m\right) = \sum_{i=0}^{m} (mkw_{norm}(t_i) + sn + kc) - sn, \tag{4}$$

where $mkw_{norm}(t_i)$ denotes the minimum keystrokes of word t_i using a general input method,

$$mkw_{norm}(t_i) = len(t_i) \times rk, \tag{5}$$

where $len(t_i)$ is the character number of t_i.

(2) The count of empirical keystrokes using the MT-based input method with the MT system,

$$ek\left(Q, t_1^m\right) = \min_{cp_1^q \in CP(t_1^m)} \left\{ \sum_{i=1}^{q} \left(ek(Q, cp_i) + sn\right) - sn \right\}, \tag{6}$$

where $CP(t_1^m)$ denotes the set of all partitions, each of which breaks the human translation t_1^m into non-empty contiguous sub-sequences; cp denotes a specific partition member of set $CP(t_1^m)$; q refers to the number of sub-sequences in cp; and $ek(Q, cp_i)$ denotes the empirical keystrokes to input the sub-sequence cp_i. Let P denote the n-gram prediction list; then, $ek(Q, cp_i)$ can be defined as

$$ek(Q, cp_i) = \begin{cases} kc & cp_i \in P \\ len(t_i) + kc & cp_i \notin P, cp_i \in Q \\ mkw_{norm}(cp_i) + kc & cp_i \notin P, cp_i \notin Q \end{cases}. \tag{7}$$

(3) Given the maximum length of n-gram prediction list W, the count of minimum idealized keystrokes using the MT-based input method with the aid of the perfect MT candidate t_1^m is

$$pk\left(t_1^m\right) = \begin{cases} \frac{m}{W} \times (kc + sn) - sn & m \bmod W = 0 \\ \lfloor \frac{m}{W} \rfloor \times (kc + sn) + kc & m \bmod W \neq 0 \end{cases}. \tag{8}$$

The default value of W is set to 4 in this article.

In conclusion, with the above sufficient statistics, the value of MinKSR at the sentence level can be calculated by Equation (3). In addition, the value of MinKSR at the corpus level is given by the following equation:

$$r = \frac{\sum_{t \in T} mk_{norm}(t) - \sum_{t \in T, Q \in \{Q\}} ek(Q, t)}{\sum_{t \in T} mk_{norm}(t) - \sum_{t \in T} pk(t)}, \tag{9}$$

where T denotes the set of translation references.

Both $pk(t_1^m)$ and $ek(Q, t_1^m)$ emphasize the n-gram matching. MinKSR optimizes the deep information required by the CAT-oriented input method, including n-best candidates, hypotheses, translation rules, and the perfect beginnings of the final MT results.

Moreover, following Equation (5), MinKSR can be adjusted to suit other languages by changing the value of rk according to Cui (1985) and Garay-Vitoria and Abascal (2006).

3.2 MinKSR with Length Penalty

We have considered the word choice and the word order in the baseline MinKSR. We now focus on the length of MT candidates. Let c be the average length of the final MT candidates in the n-best list and t be the reference length. Inspired by BLEU, we compute the brevity penalty BP:

$$BP = \begin{cases} 1 & if\, c \leq t \\ e^{1-\frac{c}{t}} & if\, c > t \end{cases}. \tag{10}$$

Then,

$$MinKSR = BP \times r. \tag{11}$$

The value of MinKSR ranges from 0 to 1. The higher the value of MinKSR is, the more keystrokes that can be saved. The *perfect* MT result for the source sentence will attain a score of 1 if it is identical to the human translation.

The actual editing interface of the CoCat input method, incorporating with the CoTrans Translator platform, is shown as Figure 5. The source sentences are displayed in Zone A, and the current sentence is highlighted for identification. Simultaneously, the corresponding machine translation result, guided by the MinKSR evaluation metric, is shown in Zone B. We can type the expected human translation result (HT) into the text box located in Zone C, and the typing history is demonstrated by Zone D1-D7. As we can see, the phrase generation model works smoothly in the steps indicated by Zones D1, D4, and D7, and the n-gram prediction model works well at the moment indicated by Zones D2, D3, D5, and D6. In this way, the CAT-oriented input method provides effective interactions and reduces the keystrokes to improve the productivity of human translation even if the MT results are not good enough.

4 EXPERIMENTS

We conduct experiments to test the performance of the CoCat input method and the MinKSR evaluation metric in improving the productivity of human translators. To have a comprehensive understanding, we measure the human productivity from three perspectives: translation time, keystrokes, and translation quality.

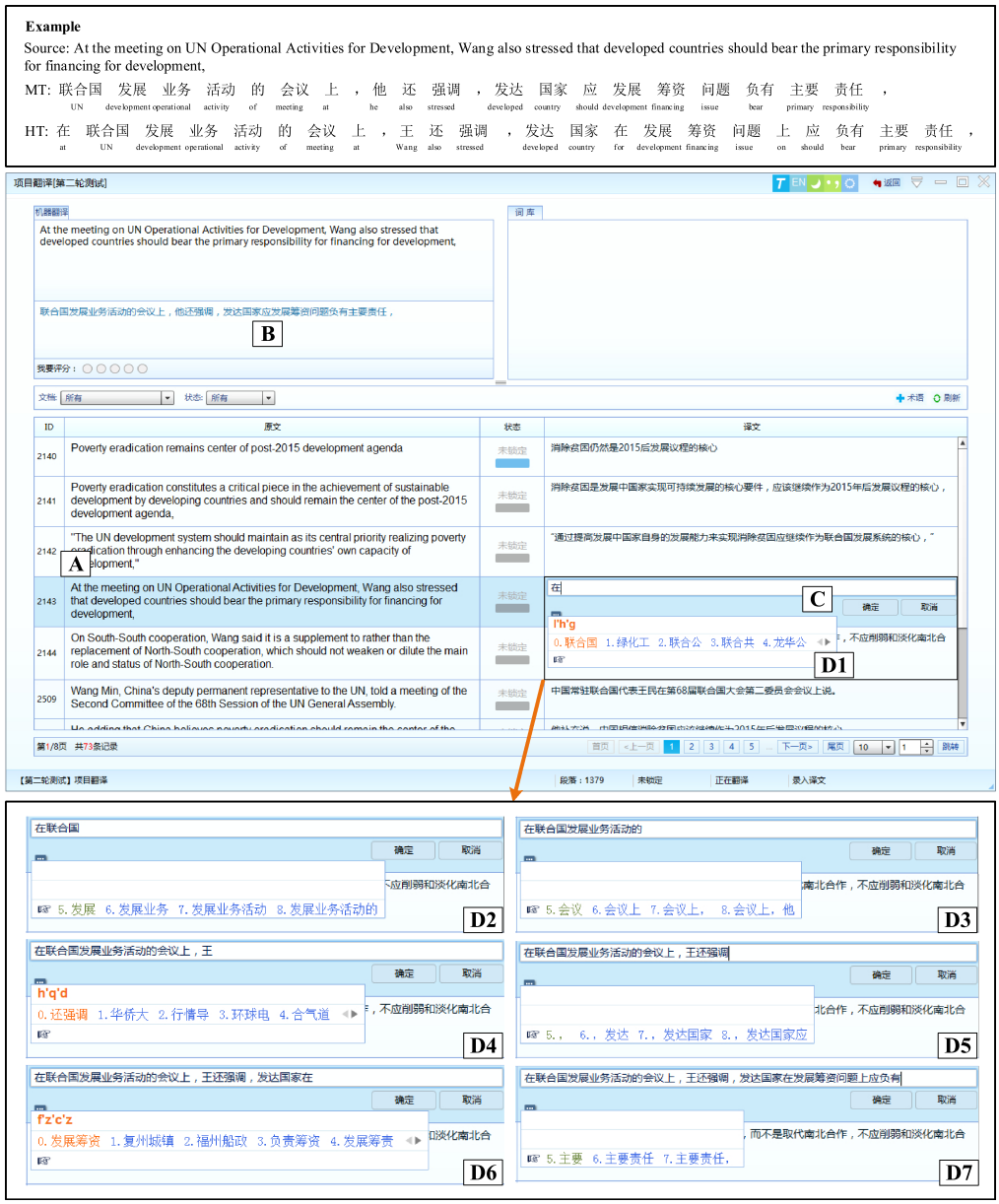


Fig. 5. The actual editing interface of the CoCat input method.

4.1 Experimental Setup

All the experiments are conducted on our CoTrans Translator platform, which is an in-house developed CAT tool integrated with a typical phrase-based MT system (Xiong et al. 2006), as shown in Figure 5. This CAT tool supports translation among many languages. We test our method on English-to-Chinese translation. The integrated MT system is trained on approximately 10,000,000 parallel sentence pairs of English–Chinese news, and it is tuned on 1,000 parallel sentence pairs

Table 1. The Statistics of the Four Groups of Human Translation Test Data M_1–M_4

English–Chinese			
#Translators	12		
Male/Female	6/6		
Test data (words)			
	BLEU	TER	MinKSR
Total	3,918	3,849	4,102
M_1	990	1,031	1,058
M_2	983	966	1,012
M_3	969	980	1,025
M_4	976	882	1,007

Note: Each group of test data contains three subsets, and each subset contains 40 sentences for the corresponding metric.

using ZMERT (Zaidan 2009) with the objective of optimizing TER (Snover et al. 2006). This tuning set, which was translated into Chinese by professional translators, was chosen from Chinese news (prior to March 2014) of China Daily. All the knowledge contained in this MT system is utilized in our proposed CoCat input method. The statistical significance test is performed using the re-sampling approach (Koehn 2004).

The CoTrans Translator platform tracks every keystroke and mouse click of the user and generates a user interaction log, which subsequently allows us to analyze the users' translation time, keystrokes, and translation quality in detail.

Next, we will introduce the participating practitioners and the experimental data.

Professional Translation Practitioners

Following the convention, we recruited 12 professional translators for our study. We evenly divided the 12 translators into 4 groups (A/B/C/D). Each translator translated the same set of sentences from English to Chinese. All of the professional translators are native Chinese speakers.

Human Translation Experimental Data

We choose 480 sentences, $S = \{s_i | i = 1, 2, \ldots, 160\}$, from China news (prior to December 2014) of China Daily as the test set for human translators. This test set contains 11,869 English words. Each sentence ranges from 23 to 26 words.

The professional translators were asked to translate the text with four different assistant tools: (1) Google Pinyin ("Google"), (2) CoCat input method ("CoCat"), (3) post-editing with Google Pinyin ("PE+Google"), and (4) post-editing with CoCat input method ("PE+CoCat"). Naturally, for each human translator, he/she should translate different sentences when using different assistant tools. Thus, we split the test data into 12 subsets randomly and evenly. Table 1 shows the details about the statistics of the 4 groups of test subset data. In Table 1, each subset includes 40 sentences for one metric. All the translators in the same groups run the exact same test. Table 2 shows the details about the permutation of assignments inspired by the previous works (Koehn 2009a; Green et al. 2014).

In the real world, there are many factors that may influence our experimental results, such as the different difficulties of the sentences to be translated, the tolerance of the long period of the translation test, and different levels of translators. To eliminate the irrelevant effects, we use

Table 2. The Permutation of Assignments for Each Assistant Tool

	A	B	C	D
Google	M_1	M_4	M_3	M_2
CoCat	M_2	M_1	M_4	M_3
PE+Google	M_3	M_2	M_1	M_4
PE+CoCat	M_4	M_3	M_2	M_1

Note: Translation subsets M_1–M_4 are assigned to the human translator groups A–D under the various assistances.

the permutation of assignments in Table 2 based on the following assumptions: (1) The minor discrepancy of difficulty degrees of four test subsets can be negligible, and (2) the fatigue degree difference of a particular translator at different times in one day can be negligible.

4.2 Data Cleaning

To exclude the translation-irrelevant factors, such as the time spent on searching for terms (Huang et al. 2016b) and moments of rest, we process the user interaction log as follows:

(1) Remove all the interactions that are irrelevant to the assistant tools from the timeline, such as looking up the dictionary online and searching for information online.
(2) Exclude all of the time intervals lasting longer than 10s between two adjacent interactions.
(3) Select the best four from the 12 human translations as references for each source sentence, and average the scores of human translations using the BLEU evaluation metric (Papineni et al. 2002).

4.3 Data Processing

We analyze human productivity in terms of translation time, keystrokes, and translation quality. To improve the robustness, we average the result values of repeated measurements. Take the "translation time," for example. According to the permutation of assignments in Table 2, the sentence s_i in subset $M1$ has been translated by three translators in group A under the assistance of "Google." For the instance s_i, we average the three values of "translation time" given by the system and obtain the value $time_{s_i}^{Google}$. We compute the average translation time of a subset under the assistance of "Google" as follows:

$$time_{M_j}^{Google} = \frac{\sum_{s_i \in M_j} time_{s_i}^{Google}}{|M_j|}, j = 1, 2, 3, 4.$$

Then, we calculate the average translation time of all sentences under the assistance of "Google" using the following formula:

$$time^{Google} = \frac{\sum_{i=1}^{160} time_{s_i}^{Google}}{160}.$$

For keystrokes and translation quality, they are calculated in the same way.

4.4 Results and Analysis

4.4.1 Input Method Tests. The detailed results of input method tests are reported in Table 3. The numbers in parentheses represent the improvement over the corresponding previous line. As shown in Table 3, individual results vary. On average, however, all human translators are faster

Table 3. Translation Time, Keystrokes, and Translation Quality Based on Different Assistant Tools

Perspective	Approach	A	B	C	D	Total
Time(s)	Google	114.68	110.67	80.39	100.30	102.38
	CoCat	89.61** (21.86%↓)	98.05** (11.40%↓)	68.05** (15.35%↓)	71.56** (28.65%↓)	**84.03**** (**17.89%↓**)
	PE+Google	64.70	52.93	83.25	71.78	66.59
	PE+CoCat	52.03** (19.58%↓)	48.34** (8.68%↓)	65.43** (21.41%↓)	66.90** (6.80%↓)	**56.63**** (**14.97%↓**)
Keystrokes	Google	209.83	236.78	168.65	184.30	204.26
	CoCat	138.41** (34.04%↓)	168.13** (28.99%↓)	93.94** (44.30%↓)	124.33** (32.50%↓)	**134.85**** (**33.98%↓**)
	PE+Google	100.66	92.24	158.13	121.81	115.75
	PE+CoCat	59.36** (41.03%↓)	63.44** (31.22%↓)	80.77** (48.92%↓)	82.11** (32.60%↓)	**69.87**** (**39.64%↓**)
Quality(BLEU)	Google	68.17	72.25	75.96	71.57	72.12
	CoCat	73.72** (8.14%↑)	78.15** (8.17%↑)	83.63** (10.10%↑)	79.64** (11.27%↑)	**78.73**** (**9.17%↑**)
	PE+Google	78.49	80.74	77.02	77.72	78.79
	PE+CoCat	81.53** (3.04%↑)	85.32** (4.58%↑)	82.43** (7.03%↑)	72.76 (4.96%↓)	**81.98**** (**3.19%↑**)

Note: "**" means the scores are significantly better than the corresponding previous lines with $p < 0.01$.

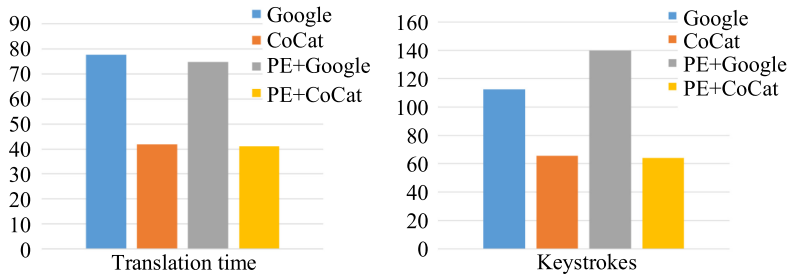


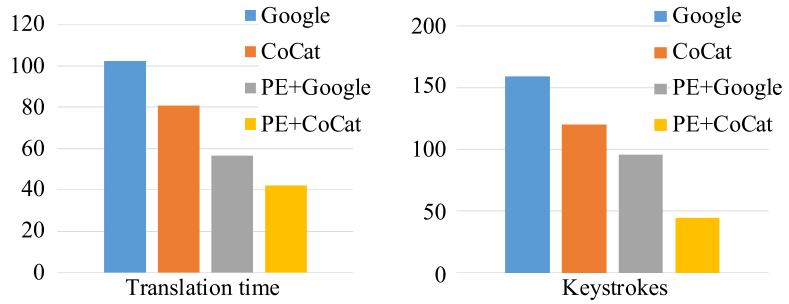
Fig. 6. The comparisons of translation time and keystrokes of the four assistances applied to a translator.

and also achieve better translation quality using any of types of assistance offered (CoCat or/plus post-editing) relative to translating from scratch with Google Pinyin. Moreover, human translators are faster and also achieve better translation quality using CoCat (translating from scratch or post-editing).

For translation time and keystrokes, lower is better. The bold figures in Table 3 show that our proposed CoCat always helps human translators significantly (with $p < 0.01$), saving more than 14% of time and over 33% of keystrokes compared with the strong baseline, i.e., post-editing using Google Pinyin (line 4 vs. line 3 and line 8 vs. line 7).

For translation quality, higher is better. The figures in Table 3 show that CoCat can also help human translators significantly improve the translation quality (with $p < 0.01$) by more than 3 absolute BLEU scores over the strong baseline.

Take a translator as an example; translation time and keystroke consumption of a specific translator in group C are reported in Figure 6. As shown in Figure 6, the CoCat helps her save approximately 46% in time and approximately 41% in keystrokes in the scratch mode and save approximately 45% in time and approximately 54% in keystrokes in the post-editing mode.



Source sentence: "CPC's discipline agency announced on Jan. 16 that Huo has been placed under investigation for suspected serious violation of party disciplines and laws".

Fig. 7. The comparisons of translation time and keystrokes of the four assistances applied to a sentence.

Take a specific sentence as an example, such as "CPC's discipline agency announced on Jan. 16 that Huo has been placed under investigation for suspected serious violation of party disciplines and laws"; the comparison statistics of translation time and keystrokes are reported in Figure 7. CoCat can save approximately 21% in time and approximately 24% in keystrokes in the scratch mode and approximately 26% in time and approximately 53% in keystrokes in the post-editing mode.

Overall, the results in Table 3 indicate that post-editing consistently outperforms unassisted translation. This result is consistent with the findings reported by Koehn (Koehn 2012). Meanwhile, that the post-editing well integrated with our proposed CoCat input method further improves the translation productivity.

Moreover, if we focus on the comparison between "CoCat" and "PE+Google," then we can find that the difference in translation quality is very small. In the industrial world, the poor performance of the automatic translation engine is often a headache for human translators to edit MT results. Fortunately, the comparison between "CoCat" and "PE+Google" indicates that we can make human translators generate better translations in less time with the aid of MT in a new way without headaches. The advantage of the novel approach is that the CoCat input method does not force human translators to revise MT results.

In summary, we can conclude that the proposed CoCat input method makes it easier for human translators to interact with MT systems effectively and imperceptibly.

4.4.2 Evaluation Metric Tests. To improve the CoCat input method, we now conduct experiments to compare MinKSR in the input method scenario with two popular metrics, BLEU and TER. The experiments include comparison tests, correlation tests, and human productivity tests.

(1) Comparison Tests

To have a general understanding about MinKSR, we first compare the test results with two popular metrics, BLEU and TER. We choose a set of 4,040 English news sentences (56,149 words) from China Daily, which was translated into Chinese (81,113 characters, 36,995 words) by professional translators, and randomly split them into two parts, e.g., the repeated experiments "Part 1" and "Part 2" in Table 4. Each part is randomly divided into two groups: a development set (Dev) including 1,000 sentence pairs, and a test set (Test) including 1,020 pairs. The integrated MT system is tuned by the corresponding development set using ZMERT (Zaidan 2009) with the objective of optimizing BLEU, TER, and MinKSR. The corresponding systems and results are denoted as "BLEU," "TER," and "MinKSR." Then, all translation results of the development sets and test sets are evaluated with BLEU, TER, and MinKSR. In addition, we count the number of sentences that

Table 4. The Comparison of BLEU, TER, and MinKSR

Part	Metric	BLEU(%)		TER(%)		MinKSR(%)		Perfect Begin.(%)	
		Dev	Test	Dev	Test	Dev	Test	Dev	Test
1	Baseline (BLEU)	**22.78**	**21.86**	**60.11**	**61.16**	**41.16**	**40.50**	41.30	40.98
	TER	21.49	20.28	58.91	60.19	39.84	38.94	38.20	36.86
	MinKSR	**22.62**	**21.91**	**60.17**	**61.23**	**41.37****	**40.73****	48.90**	48.52**
2	Baseline (BLEU)	**21.86**	**21.98**	**61.88**	**60.81**	**40.53**	**40.22**	40.60	40.39
	TER	20.00	20.40	60.33	59.89	39.15	38.84	36.10	35.69
	MinKSR	**21.59**	**22.02**	**61.49**	**60.58**	**40.75****	**40.58****	49.20**	48.73**

• Part 1 and 2 are parallel experiments.
• "**" means the scores are significantly better than the corresponding previous lines with $p < 0.05$.

Table 5. The Practical Keystroke Savings Ratio (%) Based on the MT System Tuned by BLEU, MinKSR and TER

Part	Google	CoCat-MT	CoCat(-P)+MT			CoCat(+P)+MT		
			BLEU(%)	TER(%)	MinKSR(%)	BLEU(%)	TER(%)	MinKSR(%)
1	37.40	33.93	44.20	43.28	**44.67****	48.44	47.31	**48.89****
2	36.44	35.15	45.24	44.84	**45.64****	47.70	47.04	**48.14****

• Part 1 and 2 are the repeated experiments.
• "**" means the scores are significantly better than the corresponding previous columns with $p < 0.05$.

have perfect beginning fragments, and the results are labeled "Perfect Begin." We report all the results in Table 4.

If we focus only on the bold figures (e.g., 22.78 vs. 22.62) in Table 4, then we can find that MinKSR performs very similarly with BLEU on corpus-level evaluation, whereas the difference between MinKSR and TER is considerably larger (e.g., 22.62 vs. 21.49). This result is reasonable since both MinKSR and BLEU emphasize the n-gram matching, as previously mentioned. In contrast to BLEU and TER, the figures in Table 4 show that we can increase at least 1.79 and 0.23 MinKSR scores by tuning the MT system with MinKSR on the test set. The scores show that it has the potential to reduce more keystrokes through resetting a fitting evaluation metric.

If we focus on the underlined figures in Table 4, then we can find that MinKSR can increase at least 7.5% and 10.7% of perfect beginning fragments over TER and BLEU. As previously mentioned, perfect beginning fragments are very important to the CAT scenario. Thus, the results are very significant.

To summarize, MinKSR performs very similarly to BLEU. It optimizes the intermediate MT results, such as the perfect beginning fragments, required by the CAT-oriented input method, whereas the BLEU scores of the final MT results do not really change.

(2) Correlation Tests

We further test whether the MinKSR scores are positively correlated with the practical keystroke saving ratio (PKSR) of the translation process. The translators retyped 2,040 pre-translated target (Chinese) sentences of the test set under different helper settings (a total of 8 times): the Google Pinyin input method (denoted as "Google"), the pure CoCat input method without MT ("CoCat-MT"), CoCat with the MT system but n-gram prediction disabled ("CoCat(-P)+MT"), and full-featured CoCat with the MT system ("CoCat(+P)+MT"). During the analysis, we report all the results in Table 5. The bold figures in Table 5 reveal that the CAT-oriented input method integrated with MinKSR increases over 1.10 and 0.44 PKSR scores compared to TER and BLEU on the test set.

Table 6. Translation Time, Keystrokes, and Translation Quality Based on TER, BLEU, and MinKSR

Metric	Perspective	Approach	A	B	C	D	Total
TER	Time(s)	Google	114.68	110.67	80.39	100.30	102.38
		CoCat	89.61	98.05	68.05	71.56	84.03
		PE+Google	64.70	52.93	83.25	71.78	66.59
		PE+CoCat	52.03	48.34	65.43	66.90	56.63
	Keystrokes	Google	209.83	236.78	168.65	184.30	204.26
		CoCat	138.41	168.13	93.94	124.33	134.85
		PE+Google	100.66	92.24	158.13	121.81	115.75
		PE+CoCat	59.36	63.44	80.77	82.11	69.87
	Quality(BLEU)	Google	68.17	72.25	75.96	71.57	72.12
		CoCat	73.72	78.15	83.63	79.64	78.73
		PE+Google	78.49	80.74	77.02	77.72	78.79
		PE+CoCat	81.53	85.32	82.43	72.76	81.98
BLEU	Time(s)	Google	101.48	94.67	76.45	91.45	91.32
		CoCat	76.20	78.80	61.96	62.41	69.96
		PE+Google	68.32	58.22	88.92	74.78	72.74
		PE+CoCat	51.39	50.23	50.02	68.94	55.09
	Keystrokes	Google	198.72	221.68	148.44	160.75	182.74
		CoCat	124.30	154.62	75.91	103.51	114.35
		PE+Google	104.85	96.24	163.33	131.84	124.25
		PE+CoCat	57.85	62.73	78.99	84.71	71.24
	Quality(BLEU)	Google	69.55	74.64	77.12	73.26	73.53
		CoCat	76.27	81.77	85.65	82.38	81.29
		PE+Google	77.54	81.75	75.72	74.42	77.31
		PE+CoCat	82.36	85.72	83.09	79.35	82.63
MinKSR	Time(s)	Google	109.74	102.83	78.44	101.49	98.42
		CoCat	75.08	76.24	60.44	66.14	**69.48****
		PE+Google	66.28	55.85	85.78	72.43	70.43
		PE+CoCat	46.62	46.96	61.34	63.80	**54.23****
	Keystrokes	Google	204.32	229.98	161.83	164.84	191.55
		CoCat	120.89	143.42	82.91	94.30	**110.24****
		PE+Google	102.65	95.91	160.16	125.76	121.74
		PE+CoCat	52.49	60.00	77.58	75.56	**66.48****
	Quality(BLEU)	Google	68.72	73.93	76.81	72.02	72.66
		CoCat	77.80	81.24	87.01	80.12	**81.76****
		PE+Google	77.91	82.12	75.98	76.72	78.71
		PE+CoCat	84.52	87.03	84.58	83.51	**84.32****

Note: "**" means the scores are significantly better than the corresponding previous lines with $p < 0.01$.

The correlation tests show that there is indeed a positive correlation between the MinKSR scores and the practical keystroke savings ratio.

(3) Human Productivity Tests

Finally, we test the performance of the three metrics on the ultimate goal of MinKSR, namely, improving the productivity of human translators. We analyze the human productivity in terms of translation time, keystrokes, and translation quality. To improve the robustness, we average the result values of repeated measurements. All the results are reported in Table 6. To improve clarity,

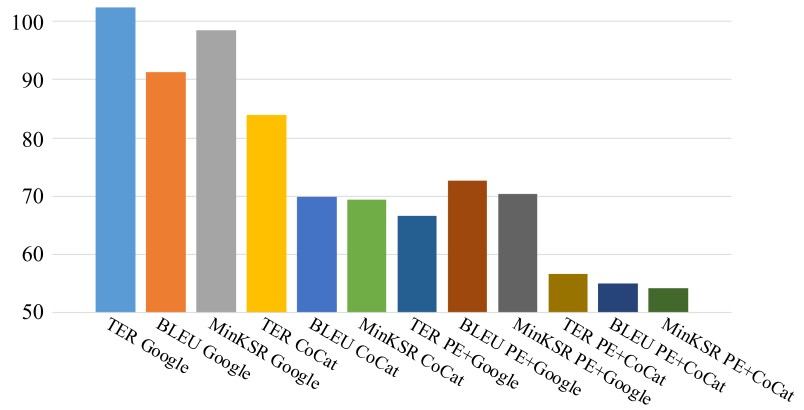


Fig. 8. The comparisons of translation time (seconds) among Term, BLEU, and MinKSR.

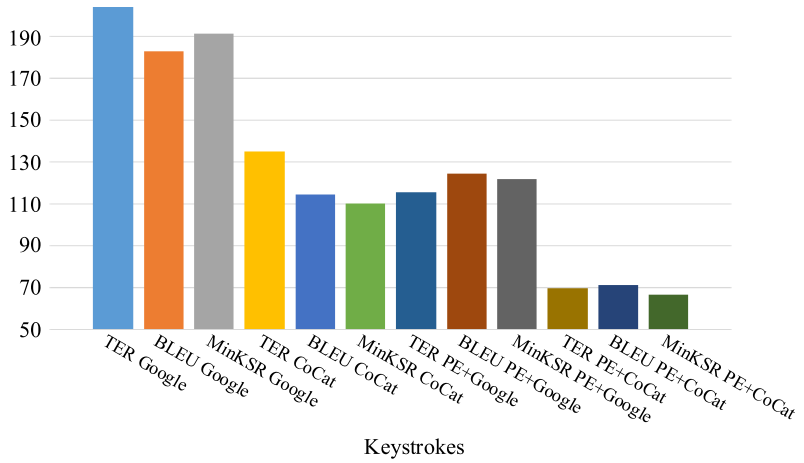


Fig. 9. The comparisons of keystrokes among Term, BLEU, and MinKSR.

the comparison statistics of translation time, keystrokes, and translation quality over various assistance are reported in Figures 8, 9, and 10, respectively. As shown in Figure 8, on average, human translators are faster and also achieve better translation quality using CoCat with MinKSR (translating from scratch or post-editing). Thus, the results in Table 6 further verified the feasibility of the MinKSR evaluation metric and the CAT-oriented input method.

For translation time and keystrokes, the bold figures in Table 6 show that our proposed MinKSR always significantly helps human translators using the CoCat input method (with $p < 0.01$), saving more than 1.59% time and over 4.85% keystrokes compared with the strong baseline (i.e., line "MinKSR PE+CoCat" vs. line "BLEU PE+CoCat" and line "MinKSR PE+CoCat" vs. line "TER PE+CoCat").

For translation quality, the figures in Table 6 demonstrate that MinKSR can also significantly help human translators using the CoCat input method improve the translation quality (with $p < 0.01$) by more than 1.6 absolute BLEU scores over the strong baseline (i.e., line "MinKSR PE+CoCat" vs. line "BLEU PE+CoCat").

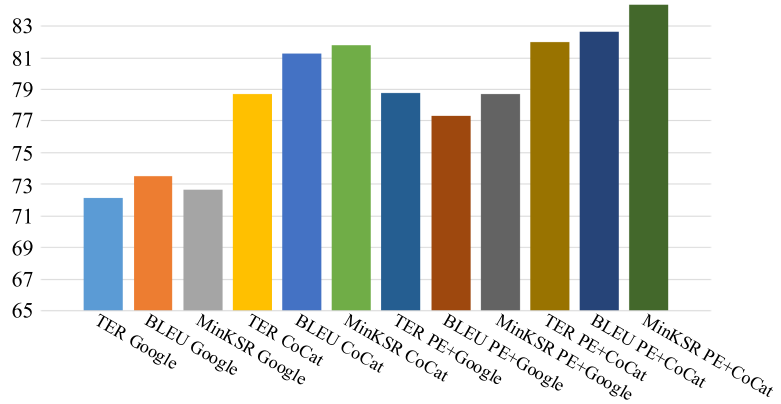


Fig. 10. The comparisons of translation quality (BLEU) among Term, BLEU, and MinKSR.

In short, the human productivity tests establish that MinKSR actually improves the productivity of human translations using the CAT-oriented input method.

In summary, the results of all the above experiments are very promising. Under the guidance of MinKSR, the underlying MT generates more useful deep information for the CAT-oriented input method. In the CAT scenario, MinKSR helps the input method substantially and firmly reduce the keystrokes of the translating process for coordinating human translators and significantly improves the actual productivity of human translations.

5 RELATED WORK

The goal of this article is to improve the productivity and efficiency of human translators by fully exploiting MT technology. The core idea is to provide human translators with translation candidates in an effectively and friendly manner. There are two types of related works that focus on offering translation suggestions.

Koehn (Koehn 2009a; Koehn et al. 2014) developed the tool Caitra, which aims at providing translation suggestions to complete the target language sentence. Based on MT post-editing, their method can offer word and phrase translation candidates through interactive machine translation. Green et al. (2014) made extensive modifications for the MT system and designed a new CAT interface. Their methods are tightly coupled with statistical machine translation in which only left-to-right decoding is allowed and dynamic decoding in interactive machine translation is generally time consuming. In contrast, we integrate most of the useful knowledge of the MT system into the CoCat input method that provides the translation suggestions in a more friendly and imperceptible manner without forcing the human translators to consider the MT outputs. In addition to using MT outputs, this is the first work to exploit deep information used by MT, such as translation rules and decoding hypotheses.

Recently, Li (2012) and Fang and Shi (2013) also attempted to incorporate the SMT information into the Chinese Pinyin input method. In their approaches, when they developed their input methods, only the MT model scores and the fuzzy word alignment between the MT output and the human translation output were employed. However, there are two disadvantages in their approaches. On the one hand, the dynamic MT model scores are difficult to calculate, and these model scores are not compatible with other features in input methods. On the other hand, the fuzzy word alignment contains a considerable amount of noise that would not substantially benefit the input

method. Rather, we design the log-linear model for the CoCat input method and integrate the translation rules, decoding hypotheses, and the n-best translation list of the MT system. In addition, we propose the n-gram prediction model to further improve the efficiency of human translators.

6 CONCLUSION

In this article, we have presented a novel input method, CoCat, which thoroughly integrates MT into CAT effectively and imperceptibly. This proposed input method is modeled with a log-linear framework and takes as features most of the useful knowledge of the MT system, such as translation rules, decoding hypotheses, and n-best translation lists. Furthermore, we have proposed a new evaluation metric to tune the underlying MT system to generate the input method preferable results. The human translation experiments on English-to-Chinese have shown that the proposed approach can not only help human translators significantly save time and keystrokes but also substantially improve the final translation quality. The experiments have also shown that post-editing well integrated with our proposed approach further improves the translation productivity.

REFERENCES

Sergio Barrachina, Oliver Bender, Francisco Casacuberta, Jorge Civera, Elsa Cubel, Shahram Khadivi, Antonio Lagarda, Hermann Ney, Jesús Tomás, Enrique Vidal, and others. 2009. Statistical approaches to computer-assisted translation. *Comput. Ling.* 35, 1 (2009), 3–28.

Michael Carl, Barbara Dragsted, Jakob Elming, Daniel Hardt, and Arnt Lykke Jakobsen. 2011. The process of post-editing: A pilot study. In *Proceedings of the 8th International NLPSC Workshop. Special Theme: Human-machine Interaction in Translation*, Vol. 41. 131–142.

Shanbo Cheng, Shujian Huang, Huadong Chen, Xinyu Dai, and Jiajun Chen. 2016. PRIMT: A pick-revise framework for interactive machine translation. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'16)*. 1240–1249.

Wei Cui. 1985. Evaluation of chinese character keyboards. *IEEE Compu.* 18, 1 (1985), 54–63.

Ruiyu Fang and Xiaodong Shi. 2013. *Research and Implementation on Aided Translation Tools Based on Input Method.* Master's thesis. Xiamen University.

George Foster and Guy Lapalme. 2002. *Text Prediction for Translators.* Ph.D. Dissertation. Université de Montréal.

Nestor Garay-Vitoria and Julio Abascal. 2006. Text prediction systems: A survey. *Univers. Access Inf. Soc.* 4, 3 (2006), 188–203.

Spence Green, Sida I. Wang, Jason Chuang, Jeffrey Heer, Sebastian Schuster, and Christopher D. Manning. 2014. Human effort and machine learnability in computer aided translation. In *Proceeding of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. 1225–1236.

Guoping Huang, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2015. A new input method for human translators: Integrating machine translation effectively and imperceptibly. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)*. 1163–1169.

Guoping Huang, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2016a. Learning from user feedback for machine translation in real-time. In *Proceedings of the 5th Conference on Natural Language Processing and Chinese Computing and the 24h International Conference on Computer Processing of Oriental Languages (NLPCC'16)*. 595–607.

Guoping Huang, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2016b. A simple, straightforward and effective model for joint bilingual terms detection and word alignment in SMT. In *Proceedings of the 5th Conference on Natural Language Processing and Chinese Computing and the 24th International Conference on Computer Processing of Oriental Languages (NLPCC'16)*. 103–115.

Guoping Huang, Chunlu Zhao, Hongyuan Ma, Yu Zhou, and Jiajun Zhang. 2016c. MinKSR: A novel MT evaluation metric for coordinating human translators with the CAT-oriented input method. In *Proceedings of the 12th China Workshop on Machine Translation (CWMT'16)*. 1–13.

Tadao Kasami. 1965. *An Efficient Recognition and Syntax Analysis Algorithm for Context-free Languages.* Technical Report Technical Report AFCRL-65-758. Air Force Cambridge Research Laboratory.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP'04)*. 388–395.

Philipp Koehn. 2009a. A process study of computer-aided translation. *Mach. Transl.* 23, 4 (2009), 241–263.

Philipp Koehn. 2009b. A web-based interactive computer aided translation tool. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP'09)*. 17–20.

Philipp Koehn. 2012. Computer-added Trasnlation. (2012). Machine Translation Marathon. http://www.mt-archive.info/MTMarathon-2012-Koehn-ppt.pdf.

Philipp Koehn, Chara Tsoukala, and Herve Saint-Amand. 2014. Refinements to interactive translation prediction based on search graphs. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL'14)*. 574–578.

Dong Li. 2012. A pinyin input method editor with english-chinese aided translation function. In *Proceedings of the 2012 International Conference on Computer Science and Service System*. 446–449.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'02)*. 311–318.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas 2006*, Vol. 200. 223–231.

Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the International Conference on Computational Linguistics and the Annual Meeting of the Association for Computational Linguistics (COLING-ACL 2'06)*. 521–528.

Daniel H. Younger. 1967. Recognition and parsing of context-free languages in time n^3. *Inf. Contr.* 10, 2 (1967), 189–208.

Omar Zaidan. 2009. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. *Prague Bull. Math. Ling.* 91 (2009), 79–88.

Ventsislav Zhechev. 2012. Machine translation infrastructure and post-editing performance at Autodesk. In *Proceedings of the AMTA 2012 Workshop on Post-Editing Technology and Practice (WPTP'12)*. 87–96.

Received November 2016; revised March 2018; accepted May 2018