ProphetMT: A Tree-based SMT-driven Controlled Language Authoring/Post-Editing Tools

Xiaofeng Wu, Jinhua Du, Qun Liu, Andy Way
ADAPT Centre, School of Computing,
Dublin City University, Ireland
{xiaofengwu,jdu,qliu,away}@computing.dcu.ie

Abstract

This paper presents ProphetMT, a tree-based SMT-driven Controlled Language (CL) authoring and post-editing tool. ProphetMT employs the source-side rules in a translation model and provides them as auto-suggestions to users. Accordingly, one might say that users are writing in a 'Controlled Language' that is 'understood' by the computer. ProphetMT also allows users to easily attach structural information as they compose content. When a specific rule is selected, a partial translation is promptly generated on-the-fly with the help of the structural information. Our experiments conducted on English-to-Chinese show that our proposed ProphetMT system can not only better regularise an author's writing behaviour, but also significantly improve translation fluency which is vital to reduce the post-editing time. Additionally, when the writing and translation process is over, ProphetMT can provide an effective colour scheme to further improve the productivity of post-editors by explicitly featuring the relations between the source and target rules.

Keywords: Controlled Language, Authoring Tool, Post-Editing, Statistical Machine Translation

1. Introduction

A Controlled Language (CL) can be defined as "an explicitly defined restriction of a natural language that specifies constraints on lexicon, grammar, and style" (Huijsen, 1998). CL is widely used in professional writing where the aim is to write for a certain standard and style demanded by a particular profession, such as law, medicine, patent, technique etc (Gough and Way, 2003; Gough and Way, 2004).

For multilingual documents, CL has been shown to improve the quality of the translation output, whether the translation is done by humans or machines (Nyberg et al., 2003).

The advantages of applying CL are self-evident: clear and consistent composition guidelines as well as less ambiguity in translation. However, the problems are also obvious: designing the rules usually requires human linguists, and rules may be difficult for end-users to grasp. In addition, the sentences that can be generated are often limited in length and complexity (O'Brien, 2003).

To the best of our knowledge, most of the existing computer-aided authoring methods (Acrolinx, for example) employ a kind of interactive paradigm with a CL together with a grammar checker which provides user feedback. Users have to follow the 'compose, check, revise' loop until the sentence is consistent according to a given parser. Exceptions usually follow an approach called conceptual authoring (Hallett et al., 2007; Hart et al., 2008) where texts are created by short cycles of language generation and user-triggered modification actions. Obviously, these methods severely restrict a user's expressiveness in the authoring process.

Even with the help of CL, current state-of-the-art machine translation (MT) methods still fail to produce reliable outputs. For example, for English-to-Chinese translation, given the sentence "allows the client computers that connect through a token ring adapter to access the network", it is very hard for computers to figure out: (i) the skeleton of this sentence is 'allow something to do something'; (ii) the boundary of the clause 'the client computers that connect ...'. Fail to parse the source sentence obviously will significantly increase the chance of producing a syntactically erroneous sentence. According to Aziz et al. (2014) and Temnikova (2012) "errors related to idiomatic expressions and word order, especially when reordering crosses phrase boundaries, seem to be connected with longer edit times " Nevertheless, these two tasks cause a human little hardship, especially to the author of the sentence.

How can we make the best of both humans and computers? The rationale underpinning ProphetMT is the following:

1. An easy-to-obtain and easy-to-learn CL which requires almost no human involvement and which has substantial expressiveness;
2. A user-friendly interface that facilitates the composition of text which is suitable (with respect to both terminal and non-terminal phrases) for an existing tree-based MT model;
3. Allows users to easily attach structural metadata while authoring;
4. The metadata can help the MT decoder to find better translations.

Finally, ProphetMT also employs a useful colour scheme to help post-editors easily visualise the relations between the source and target rules.

2. Related Work

All existing computer-aided authoring tools within a translation context employ a kind of interactive paradigm with a CL. Mitamura (1999) allows users to compose from scratch, and discusses the issues in designing a CL for rule based machine translation. Power et al. (2003) describe a CL authoring tool for multilingual generation. Marti et al. (2010) present a rule-based rewriting tool which performs syntactic analysis. Mirkin et al. (2013) introduce a confidence-driven rewriting tool which is inspired by Callison-Burch et al. (2006) and Du et al. (2010)

that paraphrases the out-of-vocabulary words (OOV) or the "hard-to-translate-part" of the source side in order to improve SMT performance.

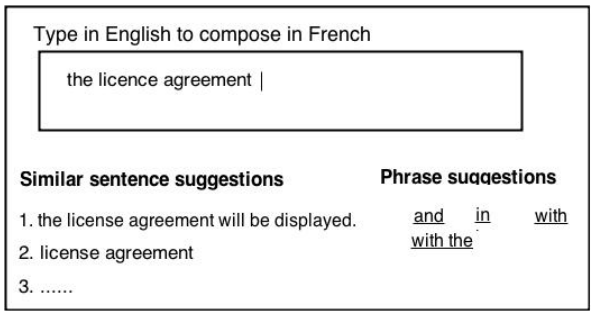

Figure 1: SMT-driven Authoring Tool by Venkatapathy and Mirkin (2012)

To the best of our knowledge, Venkatapathy and Mirkin (2012) is the first interface that could be called an SMT-driven CL authoring tool, shown in Figure 1. Their tool provides users with the word, phrase, even sentence-level auto-suggestions which are obtained from an existing translation model. Nevertheless, it lacks syntactically-informed suggestions and constraints.

Sentences in all languages contain recursive structure. Synchronous context-free grammars (SCFG) (Chiang, 2005) and stochastic inversion transduction grammars (ITG) (Wu, 1997) have been widely used in SMT and achieve impressive performance. However, MT systems which make use of SCFG tend to generate an enormous phrase table containing many erroneous rules. This huge search space not only leads to unreliable output, but also restricts the input sentence length that the system can handle. Other tree-based SMT models such as Liu et al. (2006) and Shen et al. (2008) depend heavily on the accuracy of the parsing algorithm which introduces noise upstream to the MT system.

Our method, ProphetMT, allows monolingual users to easily and naturally write correct in-domain sentences while also providing the structural metadata needed to make the parsing of the sentence unambiguous. The set of structural templates is provided by the tree-based MT system itself, meaning that highly reliable MT results can be generated directly from the user's composition.

Syntactic annotation is a tedious task which has traditionally required specialised training. In order to maintain a natural and easy writing style, ProphetMT makes use of auto-suggestion both for syntactic templates and for terms. A shift-reduce-like (Aho, 2003) authoring interface, which allows users to easily parse the "already composed part" of the sentence, is also applied to maintain the structural correctness and unambiguous parsing while the source sentence is being composed.

3. ProphetMT: A Tree-based SMT-Driven Authoring Solution

3.1. An Overview of ProphetMT

We employ the hierarchical phrase-based (HPB) translation model (Chiang, 2005) as an example. ProphetMT's main architecture is shown in Figure 2.

The main components are:

- A: the source-side auto-suggestion server, which stores the source-side rules obtained from the HPB server (component C) sorted according to their occurrence.
- B: the main UI for users to compose text. Note that we employ the 'shift-reduce' (Huang et al., 2009) manner to ensure the 'left-to-right' (or 'right-to-left', for Arabic) writing style which is more natural.
- C: modified HPB Moses server, with two main modifications: Firstly, the decoder is constrained to use the structural metadata provided by users, i.e. the parse tree is automatically constructed when the user composes the source sentence. Note that the glue-grammar (Chiang, 2005) is applied when the decoder is incapable of deriving a sound parse tree, which in our case is not an option. Therefore, the use of the glue-grammar is also prevented (i.e. the glue-grammar can be applied only at the beginning and end of the output sentence). Secondly, we set the max-span of the decoder to be unlimited (max-span= 1000).
- D: display the source-side parse tree and the synchronised target-side parse tree.
- E: the translation results and the rule alignment information produced by the HPB decoder.
- F: the post-editing auto-suggestion server which stores the target-side rules obtained from the HPB server (component C) and sorted according to their occurrence.
- G: when users focus on one of the rules (either on the tree-area or on the full-string area, explained in Figure 3), both the rule and its counterpart (in the other tree and full-string areas) will be highlighted and coloured blue. The left/right child (if present) and its descendants will be highlighted and coloured yellow, if it corresponds to the left child of the source rule; or green, if it corresponds to the right child of the source rule. (further explained in Figure 4).
- H: the post-editing UI that allows users to perform post-editing.

The main interface plus the effects of the colour scheme when hovering the mouse over one of the target rules is shown in Figure 3 and Figure 4. The four areas are:

1. the input area (upper)
2. the source-tree structural area (middle left)
3. the target-tree structural area (middle right)
4. the full-string area which shows the composed sentence and the translation (bottom)

We further define the behaviour of ProphetMT by Algorithm 1. We explain below some terminology:

- NodeBox: the recursive (nestable) editing unit
- Non-Terminal Rule (NTR): rules like "*X is X*", "*one of X*", "*has X with X*" which have variables
- Non-Terminal (NT): the "*X*" in the NTR
- Terminal Rule (TR): rules without NT

While the user is inputting text, both TR auto-completion and NTR auto-completion are provided. Auto-completion candidates are automatically selected from the normal tree-based MT model according to the guidance introduced in

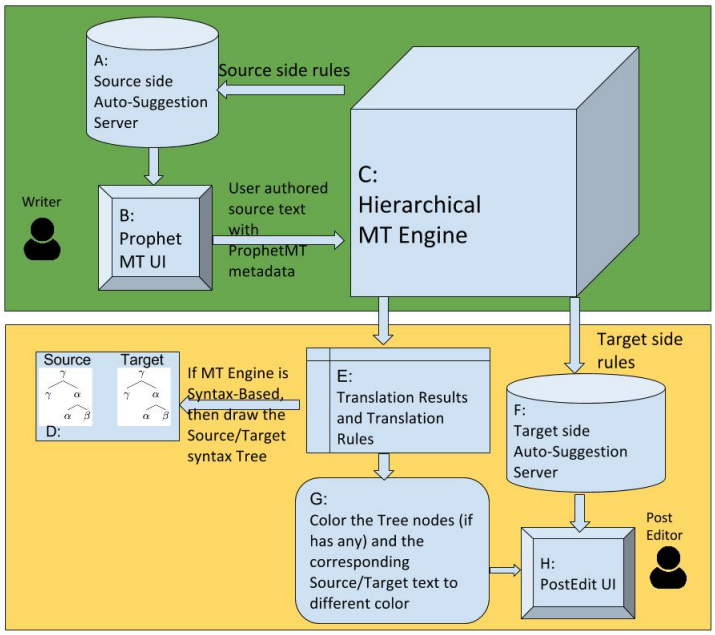

Figure 2: ProphetMT Architecture

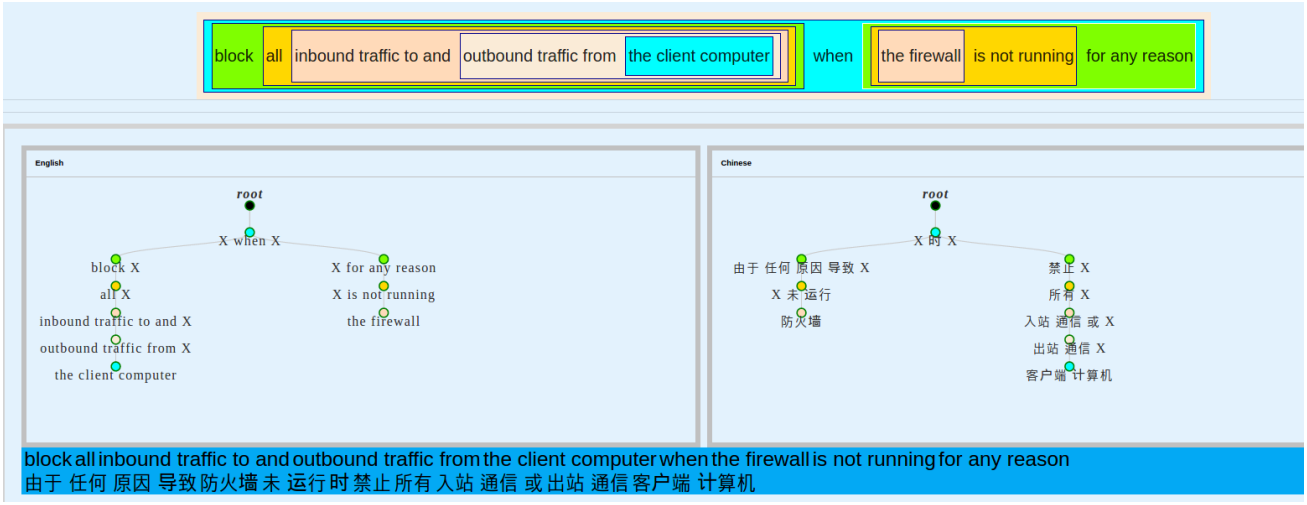

Figure 3: ProphetMT Main Interface Screenshot

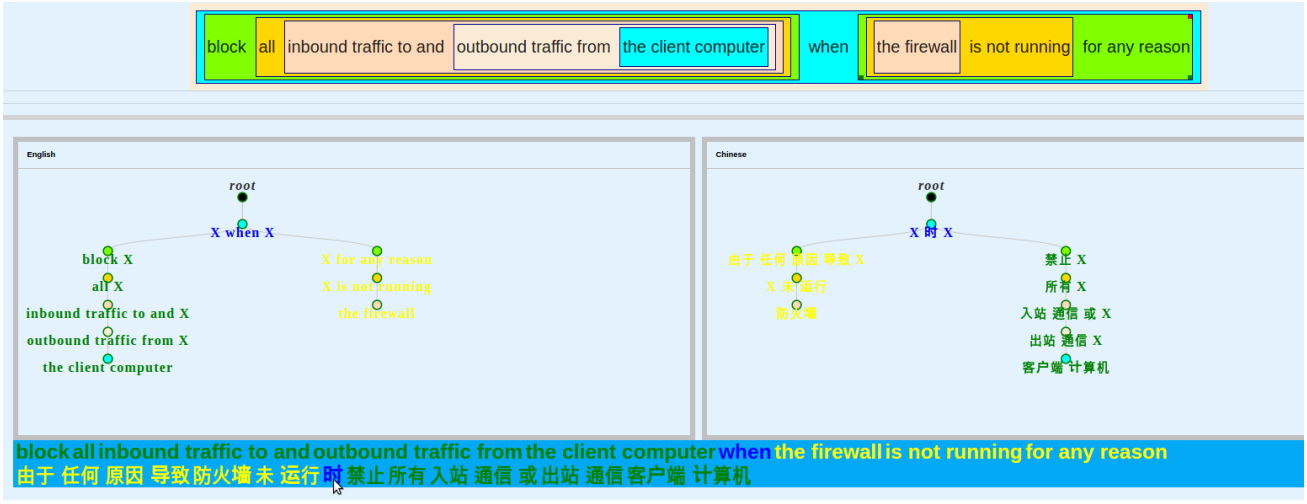

Figure 4: ProphetMT Main Interface Colour Scheme

Section 4. When the user finishes composing the sentence, the result is sent to a tree-based MT engine, and the target translation(s) are generated according to the source-side rules decided by the user.

```
Initialize: ProphetMT opens an empty NodeBox;
while User is typing in an NodeBox do
    Provide TR auto suggestions;
    if There is a left adjacent NodeBox then
        Provide all NTR suggestions;
    else
        Provide the NTRs which DO NOT have NT at the
          beginning position;
    end
    if User selects "translate" then
        Finish the source and target parse trees;
        Translate and output the results;
        Go to stop;
    end
    if User Chooses an NTR then
        Generate the according NodeBoxs;
        if The current selected NTR has an NT at the
          beginning position then
            The corresponding NodeBox will
              automatically merge with the left adjacent
              NodeBox ;
        end
        Focus goes to the first NodeBox which is empty;
        Continue;
    end
    if User starts a new NodeBox then
        Stop the current NodeBox editing;
        Focus goes to the new NodeBox;
        Continue;
    end
end
Stop:
```

Algorithm 1: ProphetMT Main Workflow

3.2. Merging

The merging process which happens in Algorithm 1 is shown in Figure 5. This merging process allows the user to compose the sentence from left-to-right while keeping the partially parsed structure intact.

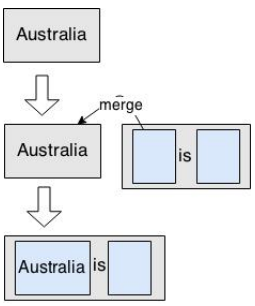

Figure 5: The Merging Process

3.3. NodeBox Starting Points Selection

Figure 6 further illustrates how the user starts a new NodeBox and how ProphetMT maintains the syntactic structure by adopting a shift-reduce-like strategy. Suppose the user has written "Australia ... and China ...". Figure 6a shows the current state in the input area and the arrows "A" and "B" are the possible insertion points. Figure 6b shows the corresponding partially parsed tree shown in the source parsing area which also indicates the two insertion positions. Figure 6c shows the parsing area when the user wants to further describe China and chooses the rule "*X which is X*" in position "A". Because there is a left-adjacent NodeBox and there is a NodeBox at the start position of the selected rule, a merging process takes place. Figure 6d shows the parsing area when the user wants to keep "Australia .. and China .." as a unit and chooses the rule "*X are X*" at position "B". As shown, a similar merging process happens.
We can see that this shift-reduce strategy allows composition to proceed from left-to-right while at the same time maintaining a correct parse of the existing text.

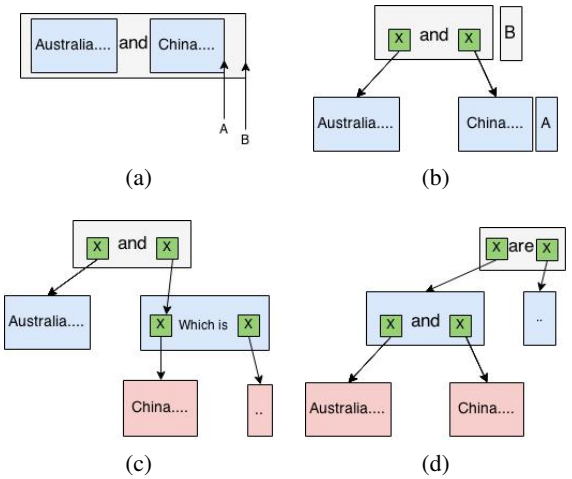

Figure 6: NodeBox Starting Points Selection

4. Auto-suggestions

In this section we introduce the auto-suggestion web-service employed in ProphetMT, including the *phrase-level*, *rule-level*, and *paraphrase* suggestion engines.

4.1. Terminal Rule (Phrase) Auto-suggestions

Following the work of Venkatapathy and Mirkin (2012), the phrase-level auto-suggestions are also guided by three factors:

- **Fluency**: What the user has input will influence the incoming auto-suggestion. We use the SRILM (Stolcke and Laboratory, 2002) toolkit to rescore the phrase auto-completion.
- **Translatability**: The phrase pairs in the phrase table (i.e. the SMT model) are sorted according to the four translation possibility features.
- **Semantic Distance**: The semantic distance of the suggested phrases must be close to the already composed part.

The final rank of the proposed phrases is based on the minimization of the Semantic Distance and maiximization of the Fluency and Translatability.

4.2. Non-Terminal Rule (NTR) Auto-suggestions

In order to extract NTRs which are meaningful to humans, we parse the source side of the training corpus with the Berkley Parser.[1] Then we wrap the parsed result with the xml for Moses[2] hierarchical phrase-based model to extract rules. The ranking of NTR suggestions follows the same methodology that was employed for phrase suggestions.

4.3. Filtering

To further reduce the amount of rules, the NTRs containing content words like nouns, pronouns and numbers can be removed. This filtering is based on the observation that the structure of a sentence is primarily dictated by function words as well as verbs. The phrase-level auto-suggestions are responsible for providing the content words that fill the leaf nodes in the hierarchical templates. Because most NTRs will be discarded, and because the source side is already parsed when feed to the decoder, the normal restrictions of tree-based models, such as the maximum span (which is $<= 20$) and the NT numbers (which is usually $<= 2$), can be removed.

4.4. Paraphrase Auto-suggestions

If the user inputs an OOV, a paraphrase engine will be queried to try to suggest terms within the current SMT model. Paraphrases are obtained from PPDB[3]. If the OOV is not found in PPDB, then the user will be forced to choose another word.

5. Experiments

This section describes the preliminary experiments conducted.

5.1. Experimental Settings

Our raw data set is the English-to-Chinese translation memory in Li et al. (2014), consisting of 86k sentence pairs. The average sentence length of the training set are 13.2 and 13.5 for English and Chinese, respectively. The development set has 762 sentence pairs and the test set has 943 sentence pairs. From the test set, we randomly select 150 sentences for our evaluation.

Our baseline is a standard HPB-SMT model with all the default settings: maximum 2 non-terminals; maximum 5 tokens for each rule; max-chart span is 10; etc. As for ProphetMT, we modify the decoder as described in Figure 2 (C).

We use the GIZA++ (Och and Ney, 2003) implementation of IBM word alignment 4, 5-gram language model with Kneser-Ney smoothing (Kneser and Ney, 1995) trained with SRILM (Stolcke and Laboratory, 2002) on the Chinese side of the training data, and Moses (Koehn et al., 2007) to decode.

The automatic evaluation metric we employ is character based 5-gram BLEU (Papineni et al., 2002), which is one of the standards in Chinese-as-target evaluation.[4]

After minimum-error-rate training (Och, 2003), the baseline obtains 49.0 BLEU score on the test set. Note that while the BLEU score might be viewed as rather high for this language pair, it is actually quite typical of the scores seen when using TMs from industry, which show much more repetition than training data used in most 'academic' MT papers.

5.2. Writing instructions

The ultimate goal of ProphetMT is to allow users to easily compose sentences in a subset of language that is 'understood' by the computer. Ideally this can be conducted by either letting different users describe the same picture or to paraphrase the same sentence. For simplicity, our experiment only requires users to rewrite the sentences in the test set using ProphetMT.

According to Wang et al. (2007), which systematically investigates English-Chinese reordering, we define the writing instructions as follows (note: words in parentheses are the expanded non-terminals):

1. verbs must be used in non-terminal rules.
2. prefer the longest phrase that composes a constituent, e.g. in Figure 3, "the firewall", "the client computer" are two phrases that act as a noun; "(the firewall) is not running" is preferable to "(the firewall) is not (running)" or "(the firewall) is (not (running))".
3. prefer a noun phrase attacheing its preposition at the right adjacent place, e.g. in Figure 3, "outbound traffic from X" is preferred to "(outbound traffic) from X"
4. prefer verb to be right-adjacent to its subject in the same rule, e.g. "something blocks X" is better than "(something) blocks X", but our experiments show that "something (blocks X)" also works fine for English to Chinese.
5. auxiliary verb must be attached with its subject as a whole rule, e.g. "something is/does X", "something is/does X"
6. relative clause should be preferred as "X that is not X" or "X that is X" ,or "X that you want to X", than "X that (is not X)" or "X that (you want to X)"
7. prefer conjunctions to glue two complete constituents; if two incomplete constituents have to be glued, the first one should not contain rightmost non-terminal, e.g. in Figure 3, the rules for "inbound traffic to and outbound traffic from" should be "(inbound traffic to) and (outbound traffic X)" or "inbound traffic to and (outbound traffic from X)"
8. set collocations are preferred to be used in a single non-terminal rule, e.g. "refrain X from X", "X can afford to X", "X fear to X", "want to X". Due to the two non-terminals settings of the current Moses system, we cannot write set collocations such as "somebody refrain somebody from doing something".
9. prefer using existing adverb + verb pattern.
10. rule out sentences containing out-of-vocabulary (OOV) words.

[1] https://code.google.com/p/berkeleyparser/

[2] http://www.statmt.org/moses/

[3] http://www.cis.upenn.edu/ ccb/ppdb/

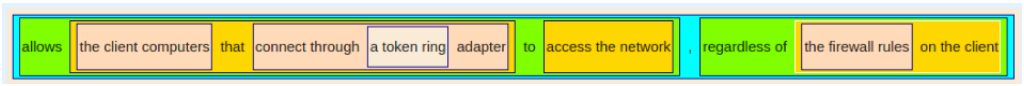

Figure 7: ProphetMT input for *allows the client computers that connect through a token ring adapter to access the network , regardless of the firewall rules on the client*

Figure 8: ProphetMT input for *authenticate to and log on to the computer that contains the symantec policy manager that was installed with the license and pre-shared secret*

Generally, if these rules cannot be satisfied, the user should rephrase this sentence to find another suitable rule to express the meaning, although this will lead to an unfair comparison with HPB model. For this experiment, the user will just choose another rule, or give up this sentence (only when an OOV occurs).
We select a subject with fluent English and ask him to use ProphetMT to rewrite the sentences in the development set. After he finishes 50 sentences, we move him to the test set.

5.3. Results and Analysis

Method	BLEU	adequacy	fluency
HPB	**49.43**	4.07	3.65
ProphetMT	49.42	**4.10**	**4.21**

Table 1: Preliminary Experimental Results.

We evaluate our final BLEU score on the selected 140 sentences (we rule out 10 sentences with OOVs) and the results are listed in Table 1. We can see that in terms of BLEU, ProphetMT does not gain any improvement. This might be due to the fact that BLEU score at such a high level cannot adequately reflect the true quality of the sentence.
To further investigate the results, we also conducted a human evaluation. Provided with references, three native Chinese speakers, denoted as A, B and C, were required to rank the first 50 of these 140-sentence outputs (HPB and ProphetMT) 1 to 5 (5 denotes the best) according to adequacy and fluency. The results also are listed in the Table 1. For these two criteria, ProphetMT outperforms HPB by 0.7% and 15.3% relatively. The average Cohen's Kappa coefficients, which are close to the upper-bounds of "moderate" and "fair", according to WMT (Bojar et al., 2014), are listed in Table 2

Evaluator	A-B	B-C	A-C	Average
adequecy	0.79	0.46	0.51	0.59
fluency	0.39	0.32	0.43	0.38

Table 2: Cohen's Kappa Coefficients between evaluators A, B, and C.

Figures 7 and 8 give two examples of the user input:
For Figure 7, the human evaluation scores are on average: 5 and 5 for adequacy and fluency vs. HPB's 4 and 4. The final output is: *yunxu tongguo lingpaihuan shipeiqi lianjie de kehuduan jisuanji fangwen wangluo , er bubi kaolv zai kehuduan shang de fanghuoqiang guize*
For Figure 8, The human evaluation scores are on average: 5 and 5 for adequacy and fluency vs. HPB's 3.3 and 2.7. The final output is: *yanzheng bing denglu baohan shiyong xukezheng he yugongxiang mima anzhuang de symantec policy manager de jisuanji*

[4]http://www.liip.cn/cwmt2013/conference.html

6. Discussions

In this paper we describe ProphetMT, which is, to the best of our knowledge, the first attempt to automatically acquire CL from a tree-based SMT model and allow authors to easily add structural metadata to help the SMT system improve. Our experiments show that using the metadata provided by ProhetMT, not only is the authoring strictly controlled within the MT model, but also significantly better SMT outputs in terms of fluency can be generated, which is very important in post-editing.
Accordingly, the value of ProphetMT lies in the following: (i) if the author is also a translator (for example: writing a bilingual contract or a CV), ProphetMT will provide the author with a SMT result promptly while he/she is writing. If the SMT output is unsatisfactory because of unsuitable source side words or results, the author can choose to modify the source side in an appropriate manner, i.e. 'to post-edit the target by post-editing the source'. (ii) if the author does not know the target language, ProphetMT will also facilitate post-editing in the following way: Firstly, the source- and target-side parse trees will be provided to the post-editor, and the source-side parse tree can be used as a 'golden standard', since it is human-parsed. Therefore, the post-editor does not need to modify the source side. Secondly, using ProphetMT's colour scheme, the post-editor can easily spot any unsatisfactorily translated rules and make modifications.
To investigate the cognitive load on the author by using ProphetMT is not a trivial task, as the controlled language provided by ProphetMT will both facilitate and control the authoring. The author might have to fall back to choose a different rule when he/she fails to find a suitable one. Nonetheless, if take into account the fact that ProphetMT does not require the user to be multilingual, even if ProphetMT slows down the author somewhat, this can be compensated by the reduced work in the post-editing stage.

7. References

Aho, A. V. (2003). *Compilers: Principles, Techniques and Tools (for Anna University), 2/e*. Pearson Education India.

Aziz, W., Koponen, M., and Specia, L. (2014). Sub-sentence level analysis of machine translation post-editing effort. *Expertise in Post-Editing: Processes, Technology and Applications.*, pages 170–199.

Bojar, O., Buck, C., Federmann, C., Haddow, B., Koehn, P., Leveling, J., Monz, C., Pecina, P., Post, M., Saint-Amand, H., et al. (2014). Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, MD, USA.

Callison-Burch, C., Koehn, P., and Osborne, M. (2006). Improved statistical machine translation using paraphrases. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 17–24, NYC. USA.

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270, MI. USA.

Du, J., Jiang, J., and Way, A. (2010). Facilitating translation using source language paraphrase lattices. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 420–429, MA, USA.

Gough, N. and Way, A. (2003). Controlled generation in example-based machine translation. In *MT Summit IX*, pages 133–140, LA, USA.

Gough, N. and Way, A. (2004). Example-based controlled translation. In *In Proceedings of the Ninth Workshop of the European Association for Machine Translation*, pages 73–81, Valetta, Malta.

Hallett, C., Scott, D., and Power, R. (2007). Composing questions through conceptual authoring. *Computational Linguistics*, 33(1):105–133.

Hart, G., Johnson, M., and Dolbear, C. (2008). *Rabbit: Developing a control natural language for authoring ontologies*. Springer.

Huang, L., Jiang, W., and Liu, Q. (2009). Bilingually-constrained (monolingual) shift-reduce parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1222–1231, Singapore.

Huijsen, W.-O. (1998). Controlled language–an introduction. In *Proceedings of International Workshop on Controlled Language Applications*, volume 98, pages 1–15, Pittsburg, Pennsylvania, USA.

Kneser, R. and Ney, H. (1995). Improved backing-off for m-gram language modeling. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, pages 181–184, MI, USA.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the association for computational linguistics on interactive poster and demonstration sessions*, pages 177–180, Prague.

Li, L., Way, A., and Liu, Q. (2014). A discriminative framework of integrating translation memory features into smt. In *the Association for Machine Translation in the Americas*, pages 249–260, Vancouver, BC, Canada.

Liu, Y., Liu, Q., and Lin, S. (2006). Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 609–616.

Marti, J., Ahs, D., Lee, B., Falkena, J., Nelson, J., Kohlmeier, B., Liger, F., Pamarthi, R., Lerum, C., Mody, V., et al. (2010). User interface for machine aided authoring and translation, May 4. US Patent 7,711,546.

Mirkin, S., Venkatapathy, S., Dymetman, M., and Calapodescu, I. (2013). Sort: An interactive source-rewriting tool for improved translation. In *Association for Computational Linguistics (Conference System Demonstrations)*, pages 85–90, Sofia, Bulgaria.

Mitamura, T. (1999). Controlled language for multilingual machine translation. In *Proceedings of Machine Translation Summit VII, Singapore*, pages 46–52.

Nyberg, E., Mitamura, T., and Huijsen, W.-O. (2003). Controlled language for authoring and translation. *Computers and Translation: A Translator's Guide*, 35:245–281.

O'Brien, S. (2003). Controlling controlled english. an analysis of several controlled language rule sets. In *Proceedings of EAMT-International Workshop on Controlled Language Applications*, pages 105–114.

Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167, Sapparo, Japan.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318, PA, USA.

Power, R., Scott, D., and Hartley, A. (2003). Multilingual generation of controlled languages. In *International Workshop on Controlled Language Applications (CLAW03)*, Dublin, Ireland.

Shen, L., Xu, J., and Weischedel, R. M. (2008). A new string-to-dependency machine translation algorithm with a target dependency language model. In *Association for Computational Linguistics*, pages 577–585, Columbus, USA.

Stolcke, A. and Laboratory, R. (2002). Srilm-an extensible language modeling toolkit. In *INTERSPEECH*, pages 901–904, Denvor, USA.

Temnikova, I. (2012). Text complexity and text simplification. *Ph.D. Thesis*.

Venkatapathy, S. and Mirkin, S. (2012). An smt-driven authoring tool. In *International Conference on Computational Linguistics (Deos)*, pages 459–466, Bombay, India.

Wang, C., Collins, M., and Koehn, P. (2007). Chinese syntactic reordering for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing-The SIGNLL Conference on Computational Natural Language Learning*, pages 737–745.

Wu, D. (1997). Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational linguistics*, 23(3):377–403.