Development of a Japanese-English Software Manual Paralell Corpus

Tatsuya Ishisaka[†]
Kazuhide Yamamoto[†]
† Nagaoka University of Technology
1603-1 Kamitomiokamachi, Nagaoka,
Niigata 940-2188, Japan
{ishisaka,ykaz}@nlp.nagaokaut.ac.jp

Masao Utiyama[††]
Eiichiro Sumita[††]
†† MASTAR Project
National Institute of Information
and Communications Tehnology
3-5, Hikaridai, Seika, Soraku,
Kyoto 619-0289, Japan
{mutiyama,eiichiro.sumita}
@nict.go.jp

Abstract

To address the shortage of Japanese-English parallel corpora, we developed a parallel corpus by collecting open source software manuals from the Web. The constructed corpus contains approximately 500 thousand sentence pairs that were aligned automatically by an existing method. We also conducted statistical machine translation (SMT) experiments with the corpus and confirmed that the corpus is useful for SMT.

1 Introduction

Multilingual parallel corpora are required to support many tasks in natural language processing. For example, statistical machine translation (SMT) requires a parallel corpus for training, and cross-lingual processing such as information retrieval and information extraction also use parallel corpora. There is no doubt on the importance of parallel corpora for any language pair.

Specially, Japanese-English parallel corpora are very scarce. Although some parallel corpora (Utiyama and Isahara, 2007) are available, the domains and sizes of these corpora are limited.

In general, European countries use multiple languages officially. Based on this multilingual environment, Koehn (2005) has built a corpus by collecting parallel texts in eleven languages from the proceedings of the European Parliament, which are published on the Web.

However, some countries such as Japan have no such language situation, that leads us difficulties for creating parallel corpora. Hence, more efforts are needed to collect them effectively.

Available Japanese-English parallel corpora are scarce. However, there are a lot of translated texts on the Web. Specially, open source manuals are translated into Japanese from English by volunteer translators.

We collected such English and Japanese texts. Then, the sentences in collected texts were automatically aligned, resulting in a parallel corpus made from open source software manuals. Manuals of open source software has been used for making a parallel corpus named *OPUS* (Tiedemann and Nygaard 2004), which was made from OpenOffice.org documentation[1], KDE manuals including KDE messages[2], and PHP manuals [3]. However, the Japanese-English part of OPUS is not large. In contrast, we collected about 500 thousand sentence pairs. In addition, our work involved extensive human efforts to ensure the quality of our parallel corpus.

The original and translated texts often proscribe copy, distribute, display, and make derivative works. Our target texts are open source software manuals. Such open source software manuals are often published under open licenses under which we can modify and distribute them.

The translation quality of open source software manuals are considered to be relatively high, because they are translated by many translators who belong to the projects and drafts of the translations are corrected by other project members. Therefore,

[1] http://www.openoffice.org

[2] http://i18n.kde.org

[3] http://www.php.net/download-docs.php

we can trust the quality of software manuals.

In the following we present how we collect, clean, and align software manuals. We also illustrate performance of SMT experiments using the corpus.

2 Target license

We will publish a parallel corpus constructed from open source software manuals. This action is considered as a redistribution with modifications. We therefore target licenses that allow redistribution and modifications.

Here are four example licenses that allow redistribution and modification.

MIT License[4] The MIT License is very open. It is necessary only to include the copyright notice and the permission notice.

FreeBSD Documentation License[5]

The FreeBSD Documentation License is similar to the MIT License. It is as follows:

> Redistribution and use in source (SGML DocBook) and 'compiled' forms (SGML, HTML, PDF, PostScript, RTF and so forth) with or without modification, are permitted provided that the following conditions are met:
>
> (1) Redistributions of source code (SGML DocBook) must retain the their copyright notice, this list of conditions and the their disclaimer as the first lines of this file unmodified.
>
> (2) Redistributions in compiled form (transformed to other DTDs, converted to PDF, PostScript, RTF and other formats) must reproduce the their copyright notice, this list of conditions and the their disclaimer in the documentation and/or other materials provided with the distribution.

Creative Commons licenses[6] Creative Commons licenses include several license types. We introduce the "Attribution-Share Alike 3.0 Unported" model in which we can copy, distribute, transmit, and adapt the work under the following conditions;

> **Attribution** You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).
>
> **Share Alike** If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

Common Development and Distribution License[7]

Common Development and Distribution License (CDDL) is a long and detailed license. Thus, we describe CDDL briefly using the FAQ of NetBeans[8].

> Modify NetBeans source code and redistribute it for free (or for sale) as long as I follow the terms of the CDDL, including the following provisions:
>
> - We must bundle the CDDL with any source code version we distribute,
> - We must make my changes to the NetBeans source code (but not new source files I create) available to the NetBeans community under the CDDL (so the community can benefit from my changes or improvements),
> - We cannot modify the rights granted under the CDDL License,
> - We can add external files to NetBeans, compile these and redistribute them for free or for sale and we do not need to make such external files or changes to them available in source code form or binary form to the NetBeans project. If my value-add is worth the price, we can sell it.

3 Characteristics of manuals

The software manuals are difficult to handle with.

Japanese open source software manuals usually contain both Japanese and English texts. Consequently, some parts of manuals are not needed for making parallel corpora. For example, when manuals explain commands, the commands are not translated into Japanese. Further, open source software manuals often include program source codes, which are not translated.

[4] http://www.opensource.org/licenses/mit-license.php

[5] http://www.freebsd.org/copyright/freebsd-doc-license.html

[6] http://creativecommons.org/licenses/by-nc-sa/3.0/deed.en

[7] http://opensource.org/licenses/cddl1.php

[8] http://wiki.netbeans.org/FaqCDDLinANutshell

Software is periodically updated, primarily by adding new features and functionality. Thus, new sentences are typically added to the corresponding manuals. As a result, the latest original document version may be newer than the translated document version. Therefore, we have to collect original and translated documents with matching versions. This needs human efforts.

In many cases, translated open source software manuals are HTML files with HTML tags that conform to the original document's format, but this need not be so. Therefore, we have to modify translated documents to match the original format. In addition, the formats of open source manuals differ from project to project. Consequently, we have to write a script to extract text portions for each project.

4 Constructing the corpus

Constructing the Japanese-English corpus takes three steps;

(1) searching for open source software manuals on the Web

(2) cleaning up documents, and

(3) aligning sentences.

We describe these three steps in the following subsections.

4.1 Searching for open source software manuals

We used Web search engines manually to search for open source software manuals. We searched for Japanese Web pages containing phrases such as 翻訳 プロジェクト (translation project). Then, we manually checked if those pages contained software manuals. If they had, we downloaded manuals. We also searched for the corresponding manuals in English. When downloading documents, we checked and matched the versions of both English and Japanese documents.

Table 1 show list of collected manuals and their URLs. In the table, JF represents "Linux Japanese FAQ Project." JF translates documents related with Linux. JM means "JM project." JM translates Linux manual pages. RFC represents "Request for Comments." Note that RFCs are not manuals, but their contents are similar to manuals and their use and importance are widespread. Others are open source software manuals.

4.2 Cleaning up documents

Software manuals contain HTML tags. We normalize documents by deleting HTML tags with a Perl script using pattern matching. This script is tailored to each software manual.

Sentences in software manuals are often broken by newlines. It is difficult to judge whether a newline character represents a sentence end or not. For example, headings are usually separated by newlines without periods. We delete newline characters in a paragraph, which is defined by a text region separated by empty lines, if that paragraph contains punctuation. Otherwise, newlines are not deleted because they are regarded as sentence ends.

4.3 Aligning sentences

We use Utiyama and Isahara's alignment method, because their method has been successfully used in aligning noisy Japanese-English parallel texts (Utiyama and Isahara, 2007). Below is a concise description of their algorithm.

We begin by obtaining the maximum similarity sentence alignments. Let J and E be a Japanese text file and an English text file, respectively. We calculate the maximum similarity sentence alignments (J_1,E_1), (J_2,E_2), $\ldots$, (J_m,E_m), using a dynamic programming matching method (Gale and Church, 1993), where (J_i, E_i) is a Japanese and English sentence alignment pair in J and E. We allow 1-to-n, n-to-1 ($0 \leq n \leq 5$), or 2-to-2 alignments when aligning sentences. The similarity between J_i and E_i is calculated based on word overlap (i.e., number of word pairs from J_i and E_i that are translations of each other based on a bilingual dictionary with 450,000+ entries). The similarity between a Japanese document, J, and an English document, E, (noted AVSIM(J,E)) is calculated using:

$$\text{AVSIM}(J, E) = \frac{\sum_{i=1}^{m} \text{SIM}(J_i, E_i)}{m} \qquad (1)$$

A high AVSIM(J,E) value occurs when the sentence alignments in J and E take on high similarity values. We also calculate the ratio of the number of

	Japanese	English
FreeBSD	http://www.freebsd.org/ja/	http://www.freebsd.org/
Gentoo Linux	http://www.gentoo.org/doc/ja/index.xml	http://www.gentoo.org/doc/en/index.xml
JF	http://www.linux.or.jp/JF/	http://www.kernel.org/pub/linux/kernel/ http://tldp.org/
JM	http://www.linux.or.jp/JM/	http://www.sfr-fresh.com/ http://www.kernel.org/ http://ftp.gnu.org/gnu/
Net Beans	http://ja.netbeans.org/index.html	http://www.netbeans.org/index.html
PEAR	http://pear.php.net/index.php	http://pear.php.net/index.php
PHP	http://www.php.net/download-docs.php	http://www.php.net/download-docs.php
PostgreSQL	http://www.postgresql.jp/	http://www.postgresql.org/
Python	http://www.python.jp/doc/	http://docs.python.org/download.html
RFC	collected from a lot of sites	http://www.rfc-editor.org/
XFree86	http://xjman.dsl.gr.jp/download.html	http://www.xfree86.org/

Table 1: List of collected manuals

sentences between J and E (noted R(J,E)) using:

$$\text{R}(J,E) = \min(\frac{|J|}{|E|}, \frac{|E|}{|J|}) \qquad (2)$$

where $|J|$ is the number of sentences in J, and $|E|$ is the number of sentences in E.

A high $R(J,E)$ value occurs when $|J| \sim |E|$. Consequently, $R(J,E)$ can be used to measure the proportion of potentially corresponding sentences. Using AVSIM(J,E) and R(J,E), we defined the similarity between J and E (noted AR(J,E)) as

$$\text{AR}(J,E) = \text{AVSIM}(J,E) \times \text{R}(J,E) \qquad (3)$$

Finally, we define the score of alignment J_i and E_i as

$$\text{Score}(J_i,E_i) = \text{SIM}(J_i,E_i) \times \text{AR}(J,E) \qquad (4)$$

A high Score(J_i,E_i) value occurs in the following case: (1) sentences J_i and E_i are similar, (2) documents J and E are similar, and (3) the number of sentences $|J|$ and $|E|$ are similar. Score(J_i,E_i) combines both sentence and document similarities to discriminate between correct and incorrect alignments.

4.4 Results of sentence alignment

We examined the results of the sentence alignment and concluded that 1-to-1, 1-to-2, or 2-to-1 sentence alignments are clean. Thus, we extracted only these sentence alignments to make our parallel corpus. Although we included all 1-to-1, 1-to-2, or 2-to-1 alignments, it is possible to extract only highly precise sentence alignments if we use the score defined in Equation 4, as verified in (Utiyama and Isahara, 2007).

Table 2 shows the number of aligned sentences. Overall, there are a total of just under 500 thousand sentences. Among these, over 90% of sentence alignments are 1-to-1.

Table 3 shows examples of aligned sentences. As the examples show, some Japanese sentences include English words, such as "PUT" or "root". We also see that both long and short sentences are included.

We found that over 80% of sentence alignments were precisely aligned. We think that further improvements are possible, since we have failed to clean up some noisy sentences. Our simple pattern match rules did not work well for removing some sentences such as notes by translators. We expect that the alignment accuracy would improve if we remove such noisy sentences.

5 MT Experiments

MT experiments were conducted to verify the usefulness of our constructed corpus for SMT. We used the Moses system (Koehn et al., 2007). We used GIZA++ (Och and Ney ,2003) for word alignment and SRILM (Stolcke,2002) for language modeling. In our experiments, we used 5-gram language models. Minimum error rate training (MERT) was per-

		English	Japanese
	sentences	tokens(average of sentences)	tokens(average of sentences)
FreeBSD	10528	156749(14.9)	245780(23.34)
Gentoo Linux	11117	1488461(13.39)	224324(20.17)
JF	122072	1867792(15.30)	2854297(23.38)
JM	41573	483098(11.62)	731045(17.58)
Net Beans	32774	450849(13.76)	682229(20.82)
PEAR	23333	294233(12.61)	446863(19.15)
PHP	67023	639857(9.55)	977281(14.58)
PostgreSQL	22843	396570(17.36)	627994(27.49)
Python	26215	297830(11.36)	499860(19.07)
RFC	128827	2229786(17.31)	3201737(24.85)
XFree86	12155	171725(14.27)	277254(22.81)
total	498460	8476950(13.77)	10768664(21.20)

Table 2: Number of aligned sentences

Japanese	English
現在設定されている PUT ファイルへのパスを含む文字列を返します。	Returns a string containing the path to the currently set put file.
これらはそれぞれ通常ユーザーと root のデフォルトパスです。	That will be a default path for normal and root users respectively.
クライアント機は Grub で、フロッピーディスクからブートします。	The client machine boots from a Grub floppy disk.
メッセージの HTTP ヘッダを含む連想配列を返します。	Returns an associative array containing the messages HTTP headers.
画像のマットチャネルを設定します。	Sets the image matte channel.
さまざまなハッシュアルゴリズムを使用して、任意の長さのメッセージに対する直接的あるいは段階的な処理を可能とします。	Allows direct or incremental processing of arbitrary length messages using a variety of hashing algorithms.
塗りつぶしや描画を行わずに現在のパスオブジェクトを終了します。	Ends current path object without performing filling and painting operations.
これはユーザが所有する BIOS 設定、カーネル構成、およびいくつかの簡素化を含んでいます。	This includes BIOS settings, kernel configuration and some simplifications in user land.
このシグナルはリモートからセッションのチェックポイントを行うときにも利用できる。	This signal can be used to perform a remote checkpoint of a session.
Xlib はテキストの描画やテキストのディメンジョンの計算で必要な時だけフォントをロードし、フォントデータをキャッシュすることを選択できる。	Xlib may choose to cache font data, loading it only as needed to draw text or compute text dimensions.

Table 3: Example of parallel sentences

formed to tune the decoder's parameters on the basis of the bilingual evaluation understudy (BLEU) score (Papinei et al., 2002). The evaluation was done using a single reference. Tuning was performed using the standard technique developed by Och (Och, 2003). The test and development data were extracted from the aligned JF sentences. Each of test and development data consists of 500 sentences.

In the following experiments, we simulated a situation where an SMT system was applied to help volunteer translators translate English JF documents into Japanese. We want to use all parallel sentences efficiently to help translators. This is a problem of domain adaptation. All of paralell sentences were translated from English to Japanese. Therefore we did MT experiments from English.

In the first experiment, we used all parallel sentences (excluding development and test sentences) as our training data, which contained approximately 500 thousand parallel sentences, as shown in Table 2. The BLEU score obtained was 37.38.

In the second experiment, we used only JF parallel sentences (approximately 100 thousand sentences). The BLEU score obtained was 40.02.

In the third experiment, we linearly interpolated language models of the first and second experiments. We changed the weight of JF's language model from 0.1, 0.3, 0.5, 0.7, and 0.9. The BLEU scores were 38.40, 39.30, 38.92, 40.07, and 42.53. The BLEU score was highest when the weight is 0.9. The translation model used was that in the first experiment.

In the fourth experiment, we log-linearly interpolated translation models of the first and second experiments. The weights were set with MERT. The BLEU score was 41.26. The language model used was that in the first experiment.

In the final experiment, we used the language model with a weight of 0.9 in the third experiment and the translation model in the fourth experiment. The BLEU score was 44.36, which was the highest in these experiments.

6 Discussion

The BLEU scores obtained were relatively high for a Japanese-English corpus. For example, Utiyama and Isahara (2007) reported a maximum BLEU score of 25.89 for patent document SMT.

However, we have to be careful about our experimental results. First, our test sentences were extracted from those having the highest alignment scores, which might not be representative samples. This was because sentences with low alignment scores could be wrong alignments, which were not suitable for measuring SMT performance. We plan to sample test sentences from the whole corpus and clean them for the purpose of evaluation in our future work. Second, our Japanese word segmenter segmented ASCII words into characters. (Japanese words were segmented properly.) For example, "word" was segmented into "w o r d" in the corpus used in the above experiments. Consequently, the BLEU scores obtained were optimistic, though the occurrences of ASCII words were much smaller than those of Japanese words.

Because the BLEU scores were rather optimistic, we manually examined test sentences. We found that short sentences were generally translated well and longer sentences were not translated well.

Overall, we concluded that our parallel corpus is useful for English-Japanese SMT developments. We also hope that this corpus will be useful for supporting human translations of these manuals.

7 Conclusion

We have reported a project on developing a Japanese-English parallel corpus made from software manuals. It has approximately 500 thousand sentence pairs. The corpus will be available at
http://www2.nict.go.jp/x/x161/members/mutiyama/manual/index.html

References

Masao Utiyama and Hitoshi Isahara. 2007. A Japanese-English Patent Parallel Corpus. In *MT summit XI*, pages 475–482.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *proceedings of the Machine Translation Summit X*,pages 79–86.

Jörg Tiedemann, Lars Nygaard. 2004. The OPUS corpus - parallel and free. In *LREC*,pages 93–96.

William A. Gale and Kenneth W. Church. 1993. A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102.

Philipp Koehn, et. al. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL Demo and Poster Sessions, pages 79–86.*

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *INSLP, pages 901–904.*

Kishore Papinei, et al. 2002. BLEU:a method for automatic evaluation of machine translation. In *ACL, pages 311–318.*

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation.In In *ACL, pages 160–167.*

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