Master's Research Project Report

Quality Improvement of Machine Translation from English to Japanese

Tomoko Ono

Supervisor Satoshi Tojo

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

June 2021

Abstract

The aim of this study is to improve the quality of machine-translated Japanese from an English source by optimizing the source content using a machine translation (MT) engine. We measured the improvement using the existing metrics, *bilingual evaluation understudy* (BLEU), and *translation edit rate* (TER) by comparing the translation from the improved source with an existing one.

We utilized the concept of *context-free grammar* (CFG) hypothesized by Noam Chomsky, an American linguist, for the translation improvement methodology: human linguistic ability is innate and can be regarded as an organ; an innate *language acquisition device* (LAD); and accordingly, every child possesses the knowledge of *universal grammar* (UG), which phrase structure rules can represent.

This theory can be applied to translation quality improvement, as the neural MT structure is similar to that of the human nerve system; however, it does not completely simulate the recognition system of the human brain. In contrast, CFG is an abstract model that simulates the human cognitive model. Therefore, it can be utilized for quality improvement in combination with a neural MT engine.

In this study, we assumed that the tree height of the CFG syntax of the source is highly correlated with the accuracy of the target translation output because of the nature of the CFG, which is considered to be an embodiment of the fundamental linguistic module representing the human mind and brain. That is, the sentence of relatively low tree height produces the accurate translation since the human cognition system can accept grammar with a simple structure.

Based on the assumption, if the pre-edited source follows the CFG phrase structure rules that possibly improve the target Japanese translation, we can then provide generic and concrete guidelines for the source content developers.

Prolog programming was used to analyze the tree height of a CFG syntax tree. Prolog is a well-known logic programming language belonging to first-order predicate logic and has a high affinity for linguistic structure analysis. For these reasons, Prolog has been used to develop artificial intelligence (AI). A Prolog program is a collection of Implication and Unit clauses. Unlike many programming languages that handle

propositional logic leading to the truth and false values using Boolean operations, such as “AND,” “OR,” and “NOT,” Prolog handles predicates that represent the state and nature of the object; for example, “A is B.” That is, Prolog holds not only Boolean values but also objects and predicates. Prolog also uses first-order predicate logic, which is a commonly applied mathematical model. For the reasons as mentioned above, it is called a logic language or logic programming language.

In this study, to script the sentence in the Chomsky normal form in Prolog, we used a definite clause grammar, for example, “`sentence :- noun_phrase, verb_phrase`” which means that a sentence consists of a noun phrase and a verb phrase. The sample code was also used in the structural analysis to determine the tree height in accordance with CFG.

Two translation quality metrics were used for the evaluation: BLEU and TER. BLEU is used as a de facto standard for the automatic calculation of translation quality. The advantage of using BLEU is to enable quality evaluation easily and immediately so that a comparison between systems can be made at a low cost without complex or manual calculations. BLEU also generates a *brevity penalty* (BP), a penalty given for extremely short sentences, resulting in a high score. The BLEU score is highly correlated with manual human evaluations. TER is a metric that measures how much the post-editor edited the machine-translated sentence to create an improved translation. The evaluation is performed by comparing a reference sentence with a machine-translated sentence. A higher score is better for BLEU, whereas a lower score is better for TER. Both metrics assume that the more similar the machine-translated sentence is to the reference sentence, the better the quality.

We collected 50 source English samples and reference translations (Reference) from the existing translation memory to conduct our experiment. The improved English source was created by polishing the original English source. Both the original and improved English sentences were input in Amazon Translate to obtain the respective Japanese translated results. Reference translation (Reference) was manually created from the original source English samples. Three types of Japanese translation results were compared with Reference: Baseline, Improved, and Postedited. Baseline is the raw result from the original source using the MT engine, Improved is from the improved English source, and Postedited is the one polished from Improved.

Next, we applied the improved English sources to CFG using Prolog, analyzed the tree height level (level), and categorized them. As a result, level 3: eight samples; level 4: 14 samples; level 5: 19 samples; level 6: five samples; level 7: two samples; level 8: two samples were obtained. The most frequently occurring level was level 5; the next most frequent was level 4.

Analysis was performed on three groups: levels 3 and 4, levels 3 to 5, and all levels after tokenization using Janome, a library of Python and a morpheme analysis engine, and then Pickle, a Python library for saving the data. The intention behind comparing levels 3 and 4 and levels 3 to 5 is to verify that sentences with higher tree height create an adverse result in translation quality and produce low-quality translations. This fact assumes that simple and clear source English sentences create a more accurate MT.

As a result, when comparing Improved with Baseline using level 3 and 4 groups, the BLEU score of Improved was increased by 5.8. The Improved TER score is 49.1, which is better than 65.5 of Baseline by 16.4, the most significant improvement among the three results, meaning that it produces an excellent Postedited score with relatively lesser effort. In the case of the level 3 to 5 group, the BLEU score gap between Baseline and Improved was 2.6. The TER difference between Baseline and Improved was 10.7, which is the second-best rate among the three results. The results show that a lower CFG tree height significantly produces better results with higher BLEU and lower TER scores.

From the results of this study, it can be concluded that the source tree height based on CFG is generally correlated with translation improvements, as shown in the BLEU and TER scores. This means that source sentences with simple grammar are easily and accurately translated, which could result from the common language acquisition system of humans proposed by Chomsky.

Given these facts, source content creation based on CFG is an effective method for translation with an MT engine. As demonstrated earlier, the application of this method can reduce the manual workload and the post-editing cost. This is beneficial from at least two perspectives: First, we do not have to pay a large sum of money to the language vendor. Second, the MT result can be improved without re-training the MT engine, which does not require specific technical expertise and consumes a considerable amount of computer resources. This achievement allows people not well trained in artificial intelligence to improve the MT results.

We believe that this study will promote and assist in developing effective new technologies in the future.

Contents

List of Figures

List of Tables

Chapter 1
Introduction

The aim of this study is to evaluate the quality of Japanese translation generated from a machine translation (MT) engine by comparing the translation from the improved source with an existing translation.

In 2014, a new translation method, neural machine translation (NMT), was developed by Google. This system employs deep learning using multi-layer artificial neural networks and is entirely different from conventional statistical machine translation (SMT). The quality of the translation was drastically improved by using the new method. However, even this higher level of quality is insufficient for the method to be used in most practical applications; indeed, manual post-editing remains critical for business use. Previously, re-training and re-modeling the translation engine itself were the only means for quality improvement. However, certain methods can be applied on the user side to enhance the quality of the output.

It has been found that the quality of Japanese translation output depends on the quality of the source English input. This finding has resulted in the need to determine the nature of source English that is optimal for the target Japanese translation to be of the highest accuracy.

Previous studies have found that a certain stylization of the source English before inputting the source into the MT engine can result in a better target Japanese translation. At the same time, according to the notion of *context-free grammar* (CFG) hypothesized by Noam Chomsky (N. Chomsky, 1956), the human linguistic ability is innate, and every child has an innate *language acquisition device* (LAD), and accordingly, the knowledge of *universal grammar* (UG), which can be represented by phrase structure rules.

In this study, we assume that the tree height of the CFG syntax of the source is highly correlated with the accuracy of the target translation output because of the nature of the CFG, which is considered to be an embodiment of the fundamental linguistic module representing the human mind and brain. That is, in CFG, the complexity of the grammar is represented in the syntax tree height, and the shallower the tree height, the more

accurate the NMT output is. It is because the neural system of the MT engine is similar to that of the human brain. However, the NMT does not completely simulate the recognition system of the human brain. In contrast, the CFG proposed by Chomsky is an abstract model that simulates the human cognitive model. Therefore, it can be utilized for quality improvement in combination with a NMT engine. Chomsky also mentioned that the study of artificial intelligence should be considered together with neurophysiology and psychology, which means my hypothesis would seem to be quite reasonable (Chomsky, Seisei Bunpou no Kuwadate (THE GENERATIVE ENTERPRISE), 2003).

To analyze the tree height of a CFG syntax tree, we used Prolog programming. Prolog is a well-known logic programming language that has been used for the development of artificial intelligence, as it belongs to first-order predicate logic and has a high affinity for linguistic structure analysis.

Importantly, suppose the pre-edited source confirming the CFG's phrase structure rules can improve the target Japanese translation. In that case, generic and concrete guidelines can be provided to the source content developers. As a result, creating such sources would meet our requirements without re-training and improving the MT engine, allowing users without technical knowledge to utilize MT optimally.

Chapter 2
Background Theories

2.1 Noam Chomsky and Context-free grammar

The reason why only human beings can acquire language ability within a short period without being taught has long been an unsettled proposition. To address this issue, Noam Chomsky, an American linguist, proposed CFG, a formal language to which human language classes can be applied. Before Chomsky, language was considered acquired after birth through the stimulation given by the mother and the child's surroundings. Linguistic studies were mainly undertaken from a liberal arts perspective and included the study of the origin of languages, such as comparative linguistics, and phonology, morphology, and syntax, such as the structural linguistics argued by Ferdinando de Saussure (Harris, 1990).

Chomsky explained that all humans have an inborn learning system within the brain, which he called a LAD (N. Chomsky, 1956). This means that language is an intrinsic human function. He also hypothesized that this ability is common across languages, and that all children have the same underlying knowledge of grammar, which he called UG (N. Chomsky, 1956). Based on this hypothesis, he also developed an advanced theory based on UG, which he labeled *generative grammar*. He regarded UG as the initial state of language acquisition and attempted to clarify how UG transitions to *generative grammar* (Chomsky, Syntactic Structures, 2002). From the perspective of linguistic studies, generative grammar mainly deals with mathematical and analytical features, that is, the generation of sentences with phrase structure rules. This notion was completely innovative and different from conventional linguistics.

The CFG, represented by the phrase structure rules, is a model to describe infinite human linguistic grammar with limited rules that develop sentences reflexively from the rules. The reason it is called "context-free" is that a sentence is not developed based on context, that is, a word's relationship with the words before/after it. The approach is very systematic and mathematical; hence, it has a strong affinity for formal languages such as programming languages.

All CFG can be represented in the Chomsky normal form (where `A`, `B`, and `C` are non-terminal symbols, `a` is a terminal symbol, and `ε` is an empty string). The Chomsky normal form is in the form of only one terminal symbol, or two non-terminal symbols on the right-hand side as shown below:

```
A → BC
A → a
S → ε
```

As described here, the CFG has only one non-terminal symbol on the left-hand side and then develops it to the right-hand side. The development continues until the non-terminal symbol is rewritten into the constituents of the terminal symbols, such as lexical items.

An example of a grammatical rule is as follows:

S → NP VP	n = it
NP → n	v = is
NP → NP PP	adj = natural
VP → v NP	n = you
PP → prep	v = are
…	adj = upset

Table 1. Grammatical Rule sample

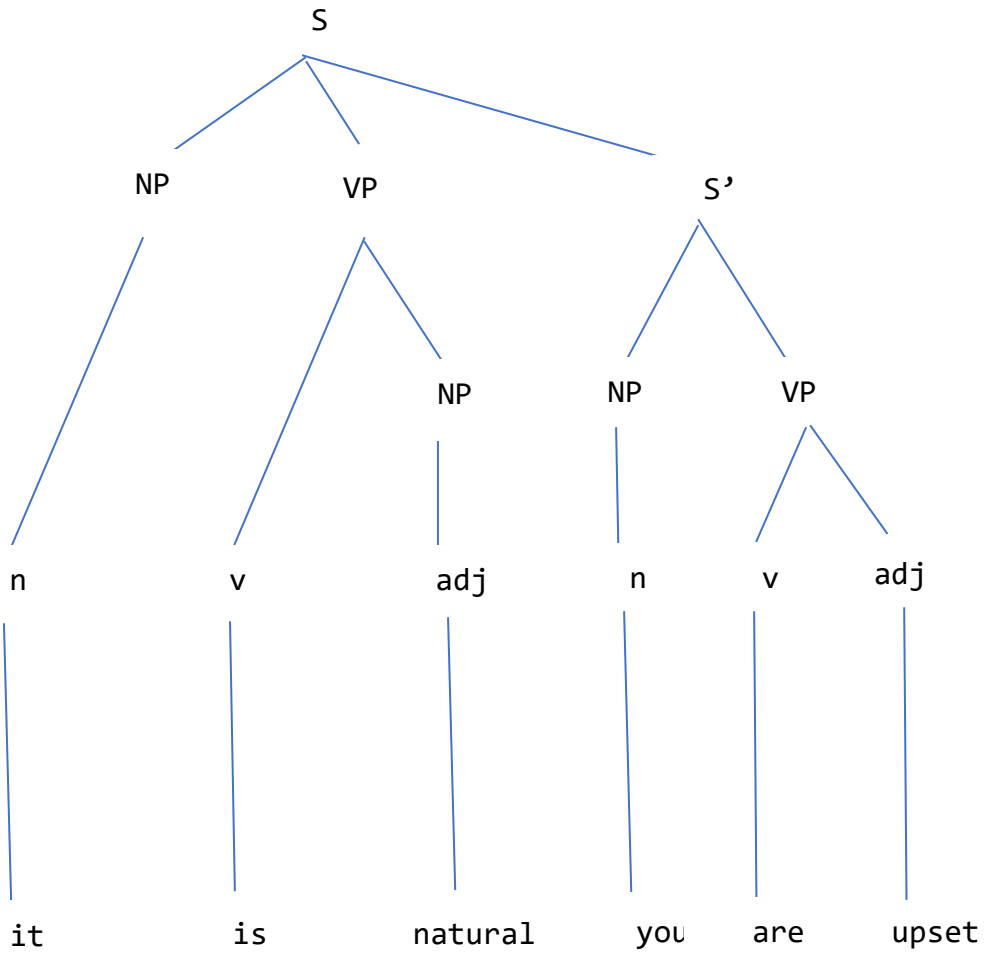

Figure 1. Syntax tree

Based on the above rule, a structure tree can be created, as shown in Fig. 1. The tree height of the trees was 4.

In some cases, multiple analysis results are generated when the sentence can be interpreted in multiple ways. However, many of these do not actually make sense.

Owing to the nature of CFG, its algorithm is easily applicable to programming languages such as Prolog. Indeed, Prolog offers an optimal way to analyze Chomsky's CFG grammar.

2.2 Prolog

2.2.1. Overview

Prolog is a programming language created by A. Colmerauer and Phillipe Roussel of Marseilles University in the early 1970s. Robert Kowalski further improved Prolog as a DEC-10 Prolog, which is ISO compliant. It is a logic programming language that uses first-order predicate logic, which is a commonly applied mathematical model. Thus, it is called a logic language or logic programming language. It is essentially a collection of propositions that indicate the relationships among the data.

Prolog is mainly used for making inferences from predicate logic. Unlike many programming languages that handle propositional logic leading to the true and false values using Boolean operations such as "AND," "OR," and "NOT," Prolog handles predicates that represent the state and nature of the object; for example, "`A is B`." That is, Prolog holds not only Boolean values but also objects and predicates.

Examples of predicate logic:

```
Fact: John is a human.
Rule: A human is mortal.
Consequence: John is mortal.
```

This can be represented as follows, where "`A`" is a variable:

```
human(John).
mortal(A) :- human(A).
```

This expression is explained in section 2.2.2 as the Implication Clause.

As explained above, the main intention of Prolog is to answer questions using the predicate format.

2.2.2. Syntax

According to H. Tanaka (Tanaka, 1989), a Prolog program consists of various terms. A term is either a constant, a variable, or a structure. Examples of a term being a constant include nouns, woman, king, 100, and others. These should not begin with an uppercase letter because some constants are reserved by the Prolog system, such as "=" and "`-->`."

Examples of a term being a variable include: `A`, `XY`, `_`, `Noun`. These should begin with an uppercase letter. Underbar "`_`" is a variable but is also a "do-not-care symbol," which works differently from the named variables for unification, as discussed later.

The structure consists of a functor and any number of attributes. An attribute must be a term. Examples of a term being a structure include:

```
s(NP,VP), np(det(Det),adj(A),n(Noun))
```

In the case of "`s(NP,VP),`" "`s`" is a functor and "`NP`" and "`VP`" are attributes. A functor must be constant and not variable.

A structure with true or false values is a predicate. A predicate that is only a functor without attributes is a preposition.

A Prolog clause is in any of the following four formats (in the examples below, `Q` and `P` are not variables but predicates.):

1) Implication clause:

```
Q :- P1, P2, …, Pn.
```

2) Unit clause:

```
Q.
```

3) Goal clause:

```
?- P1, P2, …, Pn.
```

4) Empty clause:

```
?-
```

An Implication clause is as follows:

`Q if P1, P2, … and Pn`

Or

`If P1, P2, … and Pn, then Q`

A Prolog program is a collection of Implication and Unit clauses. These are called Horn clauses. "`Q`" on the left-hand side of the symbol "`:-`" in the Implication clause is the Head; "`P1, P2, … and Pn`" on the right-hand side are the Body clauses. A Unit clause has a Body only, with no Head. Conversely, a Target clause has only a Head and no Body. An Empty clause has neither a Head nor a Body. The Body of the Target clause is called a Goal. Each `P1` of the Body is a Sub-goal. A Goal is a collection of Sub-goals. A Horn clause is the first-order predicate logic that allows only one predicate in the Head, which means that only one Head (conclusion) is confirmed when the Body (condition) is true. In this sense, the Horn clause is also called the definite clause.

2.2.3. Unification

The Prolog program performs "unification." This is a type of pattern matching performed on two items; appropriate values are assigned to the variables in either of the two items to become identical. The assignment to unify the items is called a unifier. The unifier σ for the set of two items (`ti, tj`) is shown in the example below, where σ identifies `ti` and `tj`:

Example:

When `{[a, X, end], [Y, Z, end]}, then σ= {Y=a, X=Z}` is unified successfully.

The unifier used to unify the two items was not unique. For example, in the case above, σ= {Y=a, X=Z} can also be unified. In other words, the unifier, in this case, can assign any constant to X and Z insofar as X=Z. The most common unifier among these possibilities is called the “most general unifier (mgu).”

2.2.4. DCG

In this study, to script the sentence in the Chomsky normal form in Prolog, we used *definite clause grammar* (DCG) as shown below:

```
sentence(s(NP,VP),X,Z) :-
 noun_phrase(NP,X,Y),verb_phrase(VP,Y,Z).
```

where NP is a noun phrase and VP is a verb phrase.
This means that a sentence consists of a noun phrase and a verb phrase.

Using the sample code shown in Appendix 1, we can analyze the example sentence in Figure 1 as a question (Goal clause).

```
?- sentence(X, [it, is, natural, that, you, are, upset], []).
```

The code here indicates, “What is the grammatical structure of ‘It is natural that you are upset’ ?”
Prolog returns the structure as follows:

```
s(npr(np(n(it)),  verb(vp_bv(bv(be),  npr(np(adj(natural)),
rel(np(n(you))))))), vp_bv(bv(be), np(adj(upset))))
```

This means that the sentence consists of a noun phrase (it), a “be” verb (be), an adjective (natural), and a sub sentence under the main sentence containing a noun phrase (you), a “be” verb (be), and an adjective (upset).
Thus, the analysis is successful.

In this study, as shown in Appendix 1, Prolog was used for structure analysis to determine tree height according to CFG.

2.3 BLEU and TER

Two translation quality metrics are used for evaluation: BLEU and TER.

2.3.1. BLEU

Bilingual evaluation understudy (BLEU) was proposed by Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu in 2002 (Papineni, Roukos, Ward, & Zhu, 2002).

BLEU is used as the de facto standard for the automatic calculation of translation quality. It is the official metric of various workshops used as a representative key index, as it is highly correlated with manual human evaluation results and easy to calculate by counting n-grams (in general, 4-grams are used) matching both human and MT.

The merit of using BLEU is that it produces a quality evaluation easily and immediately so that comparison among systems can be performed at a low cost without complex or manual calculations. Consequently, the development cycle can be executed at a fast pace.

The evaluation is performed by comparing a reference sentence (human translation) with a machine-translated sentence. This is based on the assumption that the more similar the machine-translated sentence is to the reference sentence, the better the quality. Possible scores are between 0 and 1, with 1 indicating the highest quality. As noted above, the BLEU score is highly correlated with manual human evaluation. The calculation is as follows: BP is the *brevity penalty*, a penalty given for sentences that are too short and often produces a deceptively high score:

$$BP = \begin{cases} 1 & if\ c > r \\ e^{(1-r/c)} & if\ c \leq r \end{cases} \quad (1)$$

Formula (1) shows the BP conditions, where

r = Length of the reference sentence
c = Length of the candidate machine-translated sentence

If the length of the candidate sentence is longer than that of the reference sentence, BP is 1 (no penalty).

If the length of the candidate sentence is shorter than that of the reference sentence, BP

is $e^{(1-\frac{c}{r})}$.

$$BLEU = BP \cdot exp\left(\sum_{n=1}^{N} w_n log P_n\right) \quad (2)$$

Formula (2) shows the expression for calculating BLEU. Here,

N = Maximum number of matched words / maximum number of n-grams
P_n = Modified n-gram precision (the expression is shown below)
w_n = Length parameter of the n-gram

P_n is calculated using Formula (3).

$$P_n = \frac{\sum_{i=1}^{S} \sum_{t_n \in h_i} min\left(count(h_{i,}t_n), max\_count(R_{i,}t_n)\right)}{\sum_{i=1}^{S} \sum_{t_n \in h_i} count(h_{i,}t_n)} \quad (3)$$

$count(h_{i,}t_n)$= Frequency of any n-gram (t_n) in a machine-translated sentence (h_i)
$max\_count(R_{i,}t_n)$= Maximum value of frequency in R_i, collection of the reference sentence of t_n

In general, $n = 4$ and $w_n = 1/n$ were used for the score measurement.
A combination of multiple n values stabilizes the resulting value.

Formula (2) can be expanded as follows:

$BLEU = BP \cdot exp\left(\sum_{n=1}^{N} w_n log P_n\right)$
where

$exp\ exp\ (x) = e^x$, *and* $w_n = \frac{1}{N}$

Then,

$exp\left(\sum_{n=1}^{N} w_n log P_n\right)$,

$= exp\left(\frac{1}{N}\sum_{n=1}^{N} log P_n\right)$

$= exp\left(\frac{1}{N}(log P_1 + log P_2 + \cdots + log P_n)\right)$

$$= exp\left(\frac{1}{N} log(P_1 \times P_2 \times \ldots \times P_n)\right)$$

$$= exp\left(log(P_1 \times P_2 \times \ldots \times P_n)^{\frac{1}{N}}\right)$$

$$= e^{log_e(log(P_1 \times P_2 \times \ldots \times P_n)^{\frac{1}{N}})}$$

$$= (P_1 \times P_2 \times \ldots \times P_n)^{\frac{1}{N}}$$

$$= \sqrt[N]{(P_1)(P_2)\ldots(P_n)}. \qquad (4)$$

As shown in Formula (4), the BLEU score is produced by multiplying BP by the Nth root of the a_i product, that is, the geometric mean.

For example,

Reference: John visited an interesting place.
Candidate: Old John visited an interesting place.

- Penalty for a shorter candidate sentence (BP): 1 (no penalty as the Candidate is longer than the Reference.)
- Match rate of mono-gram: 5/6
- Match rate of bi-gram: 4/5
- Match rate of tri-gram: 3/4
- Match rate of four-gram: 2/3

In this case, the BLEU score is

$$BLEU = \quad 1.0 \times \left(\frac{5}{6} \times \frac{4}{5} \times \frac{3}{4} \times \frac{2}{3}\right)^{\frac{1}{4}} \cong 0.76 \qquad (5)$$

In many cases, multiple candidates are provided for one reference sentence, and hence, the score is usually not as high as in this example.

However, BLEU scoring suffers from certain limitations. First, in the case of evaluating Japanese sentences, it sometimes happens that the n-gram is not well-matched, and, as a result, the BLEU score is low even though the overall meaning is the same. The

reason is due to the nature of Japanese syntax, where the case is determined by the associated particle word, not the word order and the subject can be placed anywhere in the sentence. Second, synonyms are not taken into consideration in BLEU scoring, which can produce a low score. In contrast, BLEU works well in evaluating the precision of vocabulary selection and translation fluency (Okuno, Nubig, & Masato, 2016).

Despite BLEU's limitations, there is still a high correlation to human translation for business content, as the grammar is simpler than that found in novels and other entertainment-related contexts. For the reasons above, we chose BLEU as one of the metrics for this study.

2.3.2. TER

The *translation edit rate* (TER), proposed by Matthew Snover and Bonnie Dorr in 2006 (Snover, Dorr, Schwartz, Micciulla, & Makhoul, 2006), is another measure of MT quality.

TER measures the movement of words and the number of replaced or deleted words during human post-editing. Less movement implies better translation quality. Thus, lower TER scores indicate higher quality. The calculation is shown in Equation (6).

$$TER = \frac{\#\ of\ edits}{average\ \#\ of\ reference\ words} \quad (6)$$

In general, there are four types of edits for modifying a MT by a post-editor: insertion, deletion, substitution, and shifts of single words. Every action is counted as having equal cost; moreover, shift distance is not considered.

For example,

Reference: The device is marked as lost.
Candidate: The device is reported as lost.

In this case,
Deletion: marked
Insertion: reported
Substitution: N/A
Shift: N/A

Thus, the total number of words in the Reference was six, the number of deletions was one, the number of insertions was one, and no shift or replacement action was taken.

The TER, then, can be calculated using Formula (7).

$$TER = \frac{2}{6} \cong 0.33 \quad (7)$$

Because there is only one sample, it is difficult to determine whether the above result is good or not. For accurate scoring, we would need to evaluate a collection of contexts.

According to the algorithm of Formula (7), it is clear that the lower the number of edits, the better the candidate translation quality. It is generally considered that the quality is excellent if the TER score is less than 0.3; however, this is rarely achieved in reality. A value of around 0.4 would seem to be a reasonably good score for business use.

Although not used in this study, it is worth mentioning another metric: The *human-targeted translation edit rate* (HTER). Basically, a variation of TER, HTER's algorithm is the same as TER; however, HTER evaluates the performance by creating new reference sentences using the following process:

(1) Provide reference and MT sentences to evaluators.
(2) Create new reference sentences by polishing the machine-translated sentences with reference to the reference sentences.
(3) Calculate the HTER score using the new reference sentence for the numerator and the original reference sentence for the denominator in Formula (6).

Snover showed that HTER using the new reference sentence resulted in better scores than using only the original reference sentence [Snover 06]. However, in this study, we did not use HTER because it was difficult to obtain another reference sentence for a source sentence, and due to the restriction of MultiEval, the tool used for the TER calculation.

The TER score is used more often than BLEU for business applications primarily because it clearly shows the degree of post-editing. When the TER value is low, the workload of post-editors is low, and the translation vendor can easily show cost

reduction rates to clients. In addition, it can be readily applied to the cost calculation. Easy visibility is beneficial for language vendors because they can see how much cost is reduced using the improved MT engine.

2.3.3. Other evaluation metrics considered

We considered two other evaluation metrics not included in this study: METEOR is a key metric used for the evaluation of translation quality. It is rather more generous than BLEU in terms of word selection and order in which it creates a corpus of synonyms called a “phrase table” and allows the matching of the word stem only. The main reason for not employing this score in the current study is the difficulty of creating a phrase table.

RIBES is a new method developed by Tsutomu Hirao, et al. (2014) (Hirao, et al., 2014). It is dedicated to Japanese-related evaluation, EN-JP, and JP-EN, and shows a higher correlation to human reference translations than BLEU. The method focuses on the correlation of word order from a broader perspective and allows flexibility for term variations such as METEOR. However, as it can be applied only to the Japanese language, it is not suitable for global use.

Chapter 3

Experimental Design

3.1 Preparation of samples

The data preparation and analysis process for this study are summarized below:

First, reference translations (Reference) were created from the source English samples. These are human translations that are considered correct and suitable to serve as the foundation for analyzing other MT-related sets. Three types of Japanese samples—Baseline, Improved, and Postedited—were then prepared from the source sentences for score evaluation. The intention is to compare the BLEU and TER scores to determine whether source simplification and improvement impact the machine-translated Japanese. The following procedure was used to prepare the samples:

1. Fifty source English samples (Source) were collected from the existing translation memory. These samples were obviously difficult to understand because of grammatical errors and, accordingly, resulted in poor Japanese MT. The reference translation (Reference) was manually created from Source.

2. Source was then inputted into Amazon Translate (Amazon Web Services, n.d.), a service of Amazon Web Services, to obtain raw Japanese MT results (Baseline). Amazon Translate is a general MT engine and is not customized for specific business use; consequently, objective results can be obtained. To maintain business confidentiality, some word replacements were made in Source, but no grammatical changes were made for the sake of this study.

3. Source was rewritten by simplifying and clarifying the contents. Rhetorical expressions were avoided, and straightforward and simple wording was used.

   The example below illustrates the procedure:

   Original Source: Your team will be available to help you identify what is controllable.
   Improved Source: The team will specify the things which are controllable.

   Improved Source was then inputted into Amazon Translate to obtain updated Japanese translated results. The resulting Japanese is referred to as Improved.

4. Subsequently, the new raw MT results (Improved) based on the Improved Source samples were post-edited (Postedited).

3.2 Prolog programming

The Prolog code was generated in accordance with the CFG (See Appendix 1). However, CFG is simpler than what we created for this study; for example, CFG does not allow complex sentences such as a sentence containing another sentence structure in one sentence or relative pronouns. Because such a restriction is not practical for business use, the code used in this study allows this type of structure. The code was created based on the expected use cases; however, as described, the improved sources should be simple enough to be within the scope of the definition of the code.

All 50 of the improved sentences were inputted into Prolog, and their syntax was analyzed. The answers resulting from the Prolog execution were used to validate whether tree height was correlated with an improvement in BLEU and TER scores.

Example:
Original source: You have every right to be upset.
Improved source: It is natural that you are upset.

Based on the CFG grammatical rule shown in Table 1, the analyzed result with Prolog is as follows:

```
  s(npr(np(n(it)),     verb(vp_bv(bv(be),
npr(np(adj(natural)), rel(np(n(you)))))),
vp_bv(bv(be), np(adj(upset))))
```

Subsequently, a syntax tree for each sentence was created, and the tree height was determined manually.

The syntax tree for this example is shown in Figure 1. As indicated, the tree height of the syntax tree is 4; however, it should be noted that the illustrated syntax tree height does not always match the number of parentheses designated by the Prolog analysis. This mismatch is because the definition of the code is too detailed and breaks down verbs into auxiliary verbs and general verbs.

The breakdown of the tree height levels for the 50 samples is as follows: Hereinafter, we call the tree height level simply "*level*" and employ the notion as a criterion to classify source sentences.

Level 3: 8 samples; Level 4: 14 samples; Level 5: 19 samples ; Level 6: 5 samples; Level 7: 2 samples ; Level 8: 2 samples.

The most frequently occurring level is level 5; the next most frequent is level 4.

3.3 Tokenization

Unlike European languages, in double-byte languages such as Japanese and Chinese,

the words are not separated by a single-byte space. This requires the sentences to be tokenized into morphemes for morpheme analysis before measuring the scores. For this purpose, we used Janome, a library of Python, and a morpheme analysis engine. Jacome is chosen because it is easier to install than MeCab, another common morpheme analysis tool specific to Japanese. Jacome can be installed using only the “pip” command. Although MeCab processes sentences faster than Jacome, as only 50 short sentences were analyzed in this study, differences in speed were inconsequential. Pickle, a Python library for serializing or deserializing Python objects, was used in combination with Jacome. In this case, Pickle was used to save the Python variables as a data file.

The Python code employed for the tokenization of sentences is as follows. This code tokenizes a Postedited file:

```
with open("postedited_jp2.txt", mode="r", encoding="utf-8") as f:
# Import file
postedited = f.read()

print(postedited)

import re
import pickle

postedited = re.sub("[　「」¥n]", "", postedited) # Delete single-byte and double-byte spaces and line break

separator = "。"  # Specify 。as a separator

postedited_list = postedited.split(separator)  # Split into sentences with separators

postedited_list.pop() # Delete the last element since it is empty

postedited_list = [x+separator for x in postedited_list]  # Add 。at the end of a sentence

print(postedited_list)
```

```
with open('postedited_list.pickle', mode='wb') as f:  # Save in
pickle
pickle.dump(postedited_list, f)

from janome.tokenizer import Tokenizer
import pickle

t = Tokenizer()

with open('postedited_list.pickle', mode='rb') as f:
    postedited_list = pickle.load(f)

for sentence in postedited_list:
    print(t.tokenize(sentence, wakati=True))
```

For example, when the translation of the previous English sentence (2) in Section 3.2 is tokenized, the executed result is as follows:

［’あなた’，’が’，’動揺’，’する’，’の’，’は’，’当然’，’です’，’。’］

As this format is not suitable for processing with the structure analysis tool used in this study, it was edited by deleting the single quotation marks, commas as separators, and brackets.

あなた が 動揺 する の は 当然 です 。

This preprocessing was performed on the 50 translated Japanese sentences in Reference, Baseline, Improved, and Postedited. The tokenized sentences were saved as text files in utf-8 format.

3.4 Analysis

The BLEU and TER scores were calculated for the samples generated using the methods as mentioned above. MultEval (Clark, MultEval, n.d.), a tool developed by Jonathan Clark (Clark, n.d.) of the Language Technologies Institute at Carnegie Mellon

University in 2011 using easy bootstrap resampling and approximate randomization, was used for scoring. MultEval can calculate the BLEU, TER, and METEOR scores simultaneously. However, for the reasons noted above, the METEOR score was not used in this study.

Analysis was performed on three groups: levels 3 and 4 (n=19), levels 3 to 5 (n=41), and all levels (n=50). The intention behind the comparison of levels 3 and 4 and levels 3 to 5 is to verify that sentences with higher tree heights affect translation quality and produce poorer translations. This is based on the assumption that simple and clear source English sentences create a more accurate MT.

Chapter 4

Results

4.1 Metrics results

The quality score improvement results are presented in Tables 3, 4, and 5.

All n=50	**BLEU**	**TER**
Baseline	29.1	60.8
Improved	31.6	54.9
Postedited	52.6	35.3

Table 2. Results for ‘All’ samples

Level 3 and 4 n=19	**BLEU**	**TER**
Baseline	27.5	65.5
Improved	33.3	49.1
Postedited	51.0	37.1

Table 3. Results for the sentences with levels 3 and 4

Level 3 to 5 n=41	BLEU	TER
Baseline	25.4	65.8
Improved	28.0	55.1
Postedited	49.2	38.4

Table 4. Results for the sentences with levels 3 to 5

4.2 Evaluation

We considered the Baseline and Improved results for the BLEU score, as BLEU is a metric of MT quality that is not suitable for the evaluation of the post-edited content. Postedited provides the key metric for TER comparison. Suppose the final TER score of Postedited is low and the difference between Postedited and Improved in the word order is small, and the difference between Baseline and Improved is large. In that case, it can be said that the post-editing workload is reduced owing to the source content improvement.

4.2.1. Results from 'All' samples

In all three cases, both BLEU and TER scores were considerably improved. When comparing Baseline with Improved, BLEU improved by 2.5 in the 'All' samples group for the BLEU score. The Improved TER score is 54.9, which is less (better) than the 65.5 scores for Baseline. It is obviously easier (i.e., less post-editing work is required) to use the Improved result rather than the Baseline result to obtain the Postedited TER score (35.3). The fact implies that, in general, simplification and clarification of the source content significantly affect the translation quality.

4.2.2. Results from levels 3 and 4 samples

In the case of levels 3 and 4, BLEU improved from the Baseline score of 27.5 to the Improved score of 33.3, or by a margin of 5.8. Generally, a BLEU score of over 30 indicates good quality. Thus, the quality has improved drastically. In terms of the TER score, the score difference between Baseline (65.5) and Improved (49.1) was the best improvement rate among the three results, producing a good Postedited score (37.1) involving relatively lesser effort. The sentences with level 3 and 4 tree heights were the most appropriate for MT from these results.

4.2.3. Results from levels 3 and 5 samples

In the case of levels 3 to 5, the BLEU score gap between Baseline and Improved was 3.4. The TER between Baseline (65.8) and Improved (55.1) was the second-best rate among the three results. The Postedited score (38.4) is also suitably high, as in the case of levels 3 and 4.

4.2.4. Insights

Based on these results, TER improved in all three cases, which validates the assumption that source English simplification and clarification reduce the post-editing workload. As for the correlation of CFG tree height and BLEU/TER scores, this prominently appeared in the TER and BLEU scores for the cases involving lower tree height (levels 3 and 4), as compared with the other two cases (Baseline: 27.5; Improved: 33.3).

However, in terms of the TER score comparison of ‘All’ samples and the other two results, the score of the ‘All’ samples group is the strongest counter to the assumption, whereas the score for levels 3 and 4 is better than the score for levels 3 to 5.

The following is considered to be a key reason for the result contradicting the assumption:

‘All’ samples contain several complicated and longer sentences of which tree height is higher than that of level 5, and this fact possibly contribute to the better result of ‘All’ samples than other two groups. Sentences with higher tree height tend to be longer, which means that word and phrase replacement by post-editing becomes difficult if the post-editor attempts to keep the original meaning intact as much as possible. If the post-editor tends to rewrite for better understanding, this is not “post-editing” but “rewriting.” “Rewriting” involves substantial time and cost, which is counter to the intention of MT post-editing. This could be a possible reason for the better TER rate for the ‘All’ samples group. In general; however, this can be considered as an exception because relatively short and concise sentences are often used for business, and the sentence with high tree height is less common in the business documents.

Chapter 5
Conclusion

We concluded on the basis of the results obtained in the study that the source tree height based on CFG is, in general, correlated with improvements in BLEU and TER scores. The simpler the grammatical structure, the better the BLEU/TER scores. In particular, for the samples with heights of levels 3 and 4, the quality improvement was significant.

This means that sentences with simple grammar are easily and accurately translated, which could be a result of the common language acquisition system proposed by Chomsky.

Accordingly, the possible reason for the quality improvement of the MT engine from better source English is that its “neural” network could be similar to that of the human brain. While the MT engine does not have a language acquisition system that humans have, the MT learning process is the same as that of humans insofar as each neural network has its own information but does not transfer that information to the next network until its value rises above a certain threshold. In the case of the human brain, this means that the human neural network filters the information based on the degree of interest that the person has.

Given these facts, source content creation based on the human brain structure (for example, CFG) is an effective method, even when an MT engine is used for translation.

As demonstrated above, a source English sentence with a simple grammatical structure can generate low (better) TER and high BLEU scores, which reduces the manual workload and post-editing cost. This is beneficial from at least two perspectives:

(1) From the viewpoint of localizing vendor cost reduction, a team could utilize the saved budget for possible new businesses with high potential. The workaround time would also be shorter, and a greater translation volume could be processed to improve the customer experience.
(2) With regard to the MT experience, the MT result could be improved without re-training the MT engine, requiring technology expertise and consuming a large amount

of computer resources. Rather, only polishing the source English content is required. This also results in cost reduction and is a far easier task.

To apply these results to actual businesses, source content development guidelines that restrict grammatical complications should be created.

There are two topics requiring further research: First, positive results may not be obtained solely from a simpler grammatical structure. There is a possible correlation between translation quality and the number of words in a sentence. As mentioned above, a longer sentence often consists of a complicated structure. Although we have not examined this point in this study, the topic is worth further investigation as a next step. Second, there is an exception in the case of the BLEU score. The score for the Improved in the 'All' samples group was better than that for the level 3 to 5 group, indicating that CFG tree height is not completely correlated with translation quality. This may be because a small number of samples was used. A larger number of samples should be collected for analysis to obtain more accurate data and deeper insights.

Current methods and the latest neural MT method do not consider linguistic structure, and neither developers nor users are aware of what is going on inside the engine in the analysis process, as everything is in vectorial representation.

Although the proposed method is promising, there is a ceiling for every novel technology. When the time comes, the existing and classic methods will be revisited and used in combination with a neural network approach or other future technologies. Hopefully, this study will serve to promote and assist in developing effective new technologies in this domain.

Bibliography

Amazon Web Services. (n.d.). *Amazon Translate*. Retrieved from https://aws.amazon.com/jp/translate/

Busemann, S. (n.d.). *Multidimensional Quality Metrics (MQM) Definition*. Retrieved from QT21 - Quality Translation 21: http://www.qt21.eu/mqm-definition/definition-2015-12-30.html

Chomsky, N. (2002). *Syntactic Structures.* Walter de Gruyter.

Chomsky, N. (2003). *Seisei Bunpou no Kuwadate (THE GENERATIVE ENTERPRISE).* (N. Fukui, & M. Zushi, Trans.) Tokyo: Iwanami Shoten, Publishers.

Clark, J. (n.d.). Retrieved from http://www.cs.cmu.edu/~jhclark/

Clark, J. (n.d.). *MultEval*. Retrieved from GitHub: https://github.com/jhclark/multeval

Harris, R. (1990). *Language, Saussure and Wittgenstein: How to Play Games with Words.* Psychology Press.

Hirao, T., Isozaki, H., Katsuhito, S., Duh, K., Tsukada, H., & Nagata, M. (2014). Evaluating Translation Quality with Word Order Correlations. *Shizen Gengo Shori, 21*(3), 421–44.

N. Chomsky. (1956, September). Three models for the description of language. *IRE Transactions on Information Theory, vol. 2*(no. 3), 113-124. doi:10.1109/TIT.1956.1056813

Okuno, Y., Nubig, G., & Masato, H. (2016). *Shizen Gengo Shori no Kihon to Gijutu (Basics and Technology of Natural Language Processing).* Shoeisha.

Papineni, K., Roukos, S., Ward, T., & Zhu, W. (2002). *BLEU: a method for Automatic Evaluation of Machine Translation.* ACL.

Pedregosa et al. (2011). scikit-learn Machine Learning in Python. *JMLR, 12*, 2825-2830. Retrieved from https://scikit-learn.org/stable/

scikit-learn developers. (n.d.). *Scikit-learn Machine Learning in Python*. Retrieved from 3.2.4.3.1. sklearn.ensemble.RandomForestClassifier: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html?highlight=random%20forest#sklearn.ensemble.RandomForestClassifier

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., & Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. *Proceedings of association for machine translation in the Americas*, 223–231.

Tanaka, H. (1989). *Shizen Gengo Kaiseki no Kiso (Basics of Natural Language Processing)*. Tokyo: Sangyo Tosho.

Appendix 1.

Prolog code

```
sentence(s(NP,VP),X,Z) :- noun_phrase(NP,X,Y),verb_phrase(VP,Y,Z).
noun_phrase(NP,X,Y) :- noun_phrase0(NP,X,Y).

noun_phrase(npr(NP,rel(Clause)),X,W) :- noun_phrase0(NP,X,Y),
rel_clause(Clause,Y,W).
noun_phrase(npr(NP,verb(Clause)),X,W) :- noun_phrase0(NP,X,Y),
verb_clause(Clause,Y,W).
noun_phrase(npr(NP,conj(Clause)),X,W) :- noun_phrase0(NP,X,Y),
sub_clause(Clause,Y,W).
noun_phrase(npp(NP,PP),X,Z) :-
 noun_phrase0(NP,X,Y),
 prep_phrase(PP,Y,Z).

noun_phrase0(np(n(Noun)),X,Z) :- noun(Noun,X,Z).
noun_phrase0(np(adj(A)),X,Z) :- adjective(A,X,Z).
noun_phrase0(np(n(Noun),               n(Noun2)),X,Z)              :-
noun(Noun,X,Y),noun(Noun2,Y,Z).
noun_phrase0(np(n(Noun),          n(Noun2),n(Noun3)),X,W)          :-
noun(Noun,X,Y),noun(Noun2,Y,Z),noun(Noun3,Z,W).
noun_phrase0(np(n(Noun),adv(AV)),X,Z) :- noun(Noun,X,Y),adverb(AV,Y,Z).
noun_phrase0(np(adj(A),       n(Noun),       n(Noun2)),X,W)       :-
adjective(A,X,Y),noun(Noun,Y,Z),noun(Noun2,Z,W).
noun_phrase0(np(det(Det),n(Noun)),X,Z)                            :-
determiner(Det,X,Y),noun(Noun,Y,Z).
noun_phrase0(np(det(Det),n(Noun),          n(Noun2)),X,W)         :-
determiner(Det,X,Y),noun(Noun,Y,Z),noun(Noun2,Z,W).
noun_phrase0(np(n(Noun),       det(Det),       n(Noun2)),X,W)     :-
noun(Noun,X,Y),determiner(Det,Y,Z), noun(Noun2,Z,W).
noun_phrase0(np(det(Det),adj(A),n(Noun)),X,W)                     :-
determiner(Det,X,Y),adjective(A,Y,Z),noun(Noun,Z,W).
noun_phrase0(np(adj(A),n(Noun)),X,Z)                              :-
adjective(A,X,Y),noun(Noun,Y,Z).
```

```
noun_phrase0(np(det(Det),n(Noun),adv(AV)),X,W):-
determiner(Det,X,Y),noun(Noun,Y,Z), adverb(AV,Z,W).

rel_clause(Clause,X,Z) :- relative(_,X,Y),verb_phrase(Clause,Y,Z).
rel_clause(Clause,X,Z) :- relative(_,X,Y),verb_phrase0(Clause,Y,Z).
rel_clause(Clause,X,Z) :- relative(_,X,Y),noun_phrase0(Clause,Y,Z).

verb_clause(Clause,X,Z) :- verb_phrase(Clause,X,Z).
sub_clause(Clause,X,Z) :- conjunction(_,X,Y),noun_phrase0(Clause,Y,Z).

verb_phrase(VP,X,Z) :- verb_phrase0(VP,X,Z).
verb_phrase(vpr(VP,   conj(Clause)),X,Z)   :-   verb_phrase0(VP,X,Y),
sub_clause(Clause,Y,Z).
verb_phrase0(vp(iv(IV)),X,Z) :- iverb(IV,X,Z).
verb_phrase0(vp(iv(IV),adv(AV)),X,Z) :- iverb(IV,X,Y), adverb(AV,Y,Z).
verb_phrase0(vp(tv(TV),adv(AV)),X,Z) :- tverb(TV,X,Y), adverb(AV,Y,Z).
verb_phrase0(vp(tv(TV)),X,Z) :- tverb(TV,X,Z).
verb_phrase0(vp(iv(IV),PP,VP), X,W) :-
       iverb(IV,X,Y), prep_phrase(PP,Y,Z), verb_phrase0(VP,Z,W).
verb_phrase0(vp(tv(TV),PP,VP), X,W) :-
       tverb(TV,X,Y), prep_phrase(PP,Y,Z), verb_phrase0(VP,Z,W).
verb_phrase0(vp(tv(TV),NP),X,Z) :- tverb(TV,X,Y),noun_phrase(NP,Y,Z).
verb_phrase0(vp_adv(adv(AV),tv(TV),NP),X,W) :-
       adverb(AV,X,Y),tverb(TV,Y,Z),noun_phrase0(NP,Z,W).
verb_phrase0(vp_adv(adv(AV),iv(IV)),X,Z)                              :-
adverb(AV,X,Y),iverb(IV,Y,Z).
verb_phrase0(vp(tv(TV),PP,NP),X,W) :-
       tverb(TV,X,Y), prep_phrase(PP,Y,Z),noun_phrase(NP,Z,W).
verb_phrase0(vp(tv(TV),NP,adj(A)),X,W) :-
       tverb(TV,X,Y),noun_phrase(NP,Y,Z),adjective(A,Z,W).
verb_phrase0(vp_aux(aux(Aux),vp(VP)),X,Z) :-
       auxiliary(Aux,X,Y),!, verb_phrase0(VP,Y,Z).
verb_phrase0(vp_aux(adv(AV),aux(Aux),vp(VP)),X,W) :-
       adverb(AV,X,Y), auxiliary(Aux,Y,Z), !, verb_phrase(VP,Z,W).
verb_phrase0(vp_bv(bv(BV),NP),X,Z)          :-          bverb(BV,X,Y),
noun_phrase(NP,Y,Z).
```

```
verb_phrase0(vp_bv(bv(BV),adj(A)),X,Z)        :-        bverb(BV,X,Y),
adjective(A,Y,Z).
verb_phrase0(vp_bv(bv(BV),adj(A), PP),X,W) :-
       bverb(BV,X,Y), adjective(A,Y,Z), prep_phrase(PP,Z,W).
verb_phrase0(vp_bv(bv(BV),adj(A),adv(AV)),X,W) :-
       bverb(BV,X,Y), adjective(A,Y,Z), adverb(AV,Z,W).
verb_phrase0(vp_bv(bv(BV),PP, VP),X,W) :-
       bverb(BV,X,Y), prep_phrase(PP,Y,Z), verb_phrase(VP,Z,W).
verb_phrase0(vp_bv(bv(BV),ppl(PL)),X,Z)                                  :-
bverb(BV,X,Y),pastppl(PL,Y,Z).
verb_phrase0(vp_bv(bv(BV),ppl(PL),PP),X,W) :-
       bverb(BV,X,Y),pastppl(PL,Y,Z), prep_phrase(PP,Z,W).
verb_phrase0(vp_bv(bv(BV),ppl(PL),adv(AV)),X,W) :-
       bverb(BV,X,Y),pastppl(PL,Y,Z), adverb(AV,Z,W).

prep_phrase(pp(prep(Prep),NP),X,Z)      :-      preposition(Prep,X,Y),
noun_phrase0(NP,Y,Z).
prep_phrase(pp(prep(Prep),NP,   PP),X,W)   :-   preposition(Prep,X,Y),
noun_phrase0(NP,Y,Z),prep_phrase0(PP,Z,W).
prep_phrase(pp(prep(Prep),VP),X,Z)      :-      preposition(Prep,X,Y),
verb_phrase0(VP,Y,Z).
prep_phrase(pp(prep(Prep)),X,Z) :- preposition(Prep,X,Z).
prep_phrase0(pp(prep(Prep),NP),X,Z)      :-      preposition(Prep,X,Y),
noun_phrase0(NP,Y,Z).

connector(andl,[and|X],X).

adjective(additional,[additional|X],X).
adjective(aggressive,[aggressive|X],X).
adjective(another,[another|X],X).
adjective(available,[available|X],X).
adjective(bad,[bad|X],X).
adjective(comfortable,[comfortable|X],X).
adjective(controllable,[controllable|X],X).
adjective(daily,[daily|X],X).
adjective(downloaded,[downloaded|X],X).
```

```
adjective(eligible,[eligible|X],X).
adjective(final,[final|X],X).
adjective(five,[five|X],X).
adjective(following,[following|X],X).
adjective(high,[highest|X],X).
adjective(high,[high|X],X).
adjective(lost,[lost|X],X).
adjective(low,[low|X],X).
adjective(many,[many|X],X).
adjective(medium,[medium|X],X).
adjective(missed,[missed|X],X).
adjective(much,[much|X],X).
adjective(natural,[natural|X],X).
adjective(new,[new|X],X).
adjective(no,[no|X],X).
adjective(only,[only|X],X).
adjective(open,[open|X],X).
adjective(other,[other|X],X).
adjective(printed,[printed|X],X).
adjective(refurbished,[refurbished|X],X).
adjective(related,[related|X],X).
adjective(repeated,[repeated|X],X).
adjective(responsible,[responsible|X],X).
adjective(reverse,[reverse|X],X).
adjective(safe,[safe|X],X).
adjective(secondary,[secondary|X],X).
adjective(senior,[senior|X],X).
adjective(some,[some|X],X).
adjective(sorry,[sorry|X],X).
adjective(specified,[specified|X],X).
adjective(stolen,[stolen|X],X).
adjective(stuck,[stuck|X],X).
adjective(suggested,[suggested|X],X).
adjective(ten,[ten|X],X).
adjective(three,[three|X],X).
adjective(thirty,[thirty|X],X).
```

```
adjective(thirtieth,[thirtieth|X],X).
adjective(twenty,[twenty|X],X).
adjective(uncomfortable, [uncomfortable|X],X).
adjective(upset,[upset|X],X).

adverb(also,[also|X],X).
adverb(automatically,[automatically|X],X).
adverb(clockwise,[clockwise|X],X).
adverb(counterclockwise,[counterclockwise|X],X).
adverb(everytime,[everytime|X],X).
adverb(once,[once|X],X).
adverb(patiently,[patiently|X],X).
adverb(specifically,[specifically|X],X).
adverb(twice,[twice|X],X).
adverb(up,[up|X],X).


auxiliary(can, [can|X],X).
auxiliary(donot, [donot |X],X).
auxiliary(doesnot, [doesnot |X],X).
auxiliary(may, [may|X],X).
auxiliary(might, [might|X],X).
auxiliary(mightnot, [mightnot|X],X).
auxiliary(must, [must |X],X).
auxiliary(should, [should |X],X).
auxiliary(will, [will |X],X).
auxiliary(be, [are |X],X).

bverb(be,[am|X],X).
bverb(be,[are|X],X).
bverb(be,[be|X],X).
bverb(be,[is|X],X).
bverb(be,[isnot|X],X).
bverb(be,[was|X],X).

conjunction(after,[after|X],X).
```

```
conjunction(because,[because|X],X).
conjunction(if,[if|X],X).
conjunction(than,[than|X],X).
conjunction(because,[because|X],X).
conjunction(that,[that|X],X).
conjunction(when,[when|X],X).

determiner(a,[an|X],X).
determiner(a,[a|X],X).
determiner(the,[the|X],X).

iverb(fail,[fails|X],X).
iverb(pass,[passed|X],X).
iverb(freeze,[freezing|X],X).

noun(ability,[ability|X],X).
noun(access, [access|X],X).
noun(accessory,[accessories|X],X).
noun(account,[account|X],X).
noun(action,[action|X],X).
noun(adult,[adult|X],X).
noun(alert,[alerts|X],X).
noun(amount,[amount|X],X).
noun(and,[and|X],X).
noun(app,[apps|X],X).
noun(assistance,[assistance|X],X).
noun(attribute,[attributes|X],X).
noun(badging,[badging|X],X).
noun(book,[book|X],X).
noun(button,[button|X],X).
noun(case,[case|X],X).
noun(center,[center|X],X).
noun(channel,[channel|X],X).
noun(circle,[circles|X],X).
noun(clerk,[clerk|X],X).
noun(collection,[collections|X],X).
```

```
noun(color,[colors|X],X).
noun(color,[color|X],X).
noun(concern,[concerns|X],X).
noun(condition,[condition|X],X).
noun(condition,[conditions|X],X).
noun(adult,[adult|X],X).
noun(cnfidence,[confidence|X],X).
noun(contact,[contact|X],X).
noun(contact,[contacts|X],X).
noun(conversation,[conversation|X],X).
noun(conversion,[conversion|X],X).
noun(corporate,[corporate|X],X).
noun(criteria,[criteria|X],X).
noun(customer,[customer|X],X).
noun(date,[date|X],X).
noun(day,[days|X],X).
noun(day,[day|X],X).
noun(demand,[demand|X],X).
noun(detail,[detail|X],X).
noun(device,[device|X],X).
noun(device,[devices|X],X).
noun(difficulty,[difficulty|X],X).
noun(duplication,[duplication|X],X).
noun(effort,[effort|X],X).
noun(end,[end|X],X).
noun(english,[english|X],X).
noun(expiration,[expiration|X],X).
noun(failure,[failure|X],X).
noun(family,[family|X],X).
noun(finger,[finger|X],X).
noun(finger,[fingers|X],X).
noun(folder,[folder|X],X).
noun(followup, [followup |X],X).
noun(free,[free|X],X).
noun(gap,[gap|X],X).
noun(group,[group|X],X).
```

```
noun(he,[he|X],X).
noun(here,[here|X],X).
noun(i,[i|X],X).
noun(id,[id|X],X).
noun(input,[input|X],X).
noun(issue,[issues|X],X).
noun(it,[it|X],X).
noun(item,[item|X],X).
noun(job,[jobs|X],X).
noun(knowledge,[knowledge|X],X).
noun(level,[level|X],X).
noun(lifting,[lifting|X],X).
noun(link,[links|X],X).
noun(list,[list|X],X).
noun(login,[login|X],X).
noun(manager,[manager|X],X).
noun(match,[match|X],X).
noun(matter,[matters|X],X).
noun(member,[member|X],X).
noun(membership,[membership|X],X).
noun(message,[message|X],X).
noun(mode,[mode|X],X).
noun(motor,[motor|X],X).
noun(music,[music|X],X).
noun(number,[number|X],X).
noun(not,[not|X],X).
noun(notification,[notification|X],X).
noun(object,[object|X],X).
noun(officer,[officer|X],X).
noun(one,[one|X],X).
noun(order,[order|X],X).
noun(owner,[owner|X],X).
noun(page,[page|X],X).
noun(part,[part|X],X).
noun(peak,[peak|X],X).
noun(period,[periods|X],X).
```

```
noun(person,[person|X],X).
noun(phone,[phone|X],X).
noun(priority,[priority|X],X).
noun(promotion,[promotion|X],X).
noun(purchase,[purchase|X],X).
noun(qa,[qa|X],X).
noun(quality,[quality|X],X).
noun(question,[questions|X],X).
noun(reason,[reason|X],X).
noun(refund,[refund|X],X).
noun(replacement,[replacement|X],X).
noun(representative,[representatives|X],X).
noun(satellite,[satellite|X],X).
noun(scan,[scan|X],X).
noun(screen,[screen|X],X).
noun(second,[seconds|X],X).
noun(selection,[selection|X],X).
noun(server,[server|X],X).
noun(service,[service|X],X).
noun(setting,[setting|X],X).
noun(signup,[signups|X],X).
noun(situation,[situation|X],X).
noun(someone,[someone|X],X).
noun(something,[something|X],X).
noun(space,[space|X],X).
noun(streaming, [streaming |X],X).
noun(stock,[stock|X],X).
noun(suggestion,[suggestions|X],X).
noun(support,[support|X],X).
noun(table,[tables|X],X).
noun(adult,[adult|X],X).
noun(team,[team|X],X).
noun(template,[template|X],X).
noun(their,[their|X],X).
noun(these,[these|X],X).
noun(they,[they|X],X).
```

```
noun(thing,[things|X],X).
noun(this,[this|X],X).
noun(those,[those|X],X).
noun(ticket,[tickets|X],X).
noun(time,[time|X],X).
noun(title,[titles|X],X).
noun(tolerance,[tolerance|X],X).
noun(tool,[tool|X],X).
noun(top,[top|X],X).
noun(trial,[trial|X],X).
noun(television,[television|X],X).
noun(tv,[tvs|X],X).
noun(type,[type|X],X).
noun(user,[user|X],X).
noun(users,[users|X],X).
noun(version,[version|X],X).
noun(video,[video|X],X).
noun(warranty,[warranty|X],X).
noun(we,[we|X],X).
noun(word,[words|X],X).
noun(work,[work|X],X).
noun(xbox,[xbox|X],X).
noun(you,[you|X],X).
noun(your,[your|X],X).
noun(yourself,[yourself|X],X).

pastppl(apply,[applied|X],X).
pastppl(appear,[appeared|X],X).
pastppl(liked,[liked|X],X).
pastppl(connect,[connected|X],X).
pastppl(draw, [drawn |X],X).
pastppl(read, [read|X],X).
pastppl(perform, [performed |X],X).
pastppl(deliver, [delivered |X],X).
pastppl(pass,[passed|X],X).
pastppl(process,[processed|X],X).
```

```
pastppl(deduct,[deducted|X],X).
pastppl(perform,[performed|X],X).
pastppl(ignore, [ignored |X],X).
pastppl(enter, [entered |X],X).
pastppl(do, [done |X],X).
pastppl(report, [reported|X],X).
pastppl(require, [required|X],X).

preposition(after,[after|X],X).
preposition(against,[against|X],X).
preposition(as,[as|X],X).
preposition(at,[at|X],X).
preposition(beyond,[beyond|X],X).
preposition(by,[by|X],X).
preposition(during,[during|X],X).
preposition(for,[for|X],X).
preposition(from,[from|X],X).
preposition(in,[in|X],X).
preposition(into,[into|X],X).
preposition(of,[of|X],X).
preposition(on,[on|X],X).
preposition(through,[through|X],X).
preposition(to,[to|X],X).
preposition(under,[under|X],X).
preposition(with,[with|X],X).
preposition(within,[within|X],X).
preposition(without,[without|X],X).

tverb(accept, [accept|X],X).
tverb(access, [access|X],X).
tverb(add, [add|X],X).
tverb(address, [address|X],X).
tverb(answer,[answer|X],X).
tverb(appear, [appeared|X],X).
tverb(ask, [asks|X],X).
tverb(assure, [assure|X],X).
```

```
tverb(belong, [belong|X],X).
tverb(cause, [caused|X],X).
tverb(check, [check|X],X).
tverb(connect, [connect|X],X).
tverb(contact, [contacted|X],X).
tverb(continue, [continue |X],X).
tverb(create, [create |X],X).
tverb(deduct, [deducted |X],X).
tverb(depend, [depends |X],X).
tverb(do, [did |X],X).
tverb(discuss, [discuss |X],X).
tverb(display, [display |X],X).
tverb(dissociate, [dissociate |X],X).
tverb(do, [do |X],X).
tverb(double-tap, [double-tap |X],X).
tverb(draw, [draw |X],X).
tverb(drop, [dropped |X],X).
tverb(enter, [entered |X],X).
tverb(escalate, [escalate |X],X).
tverb(feel, [feels |X],X).
tverb(file, [filed |X],X).
tverb(fill, [fill |X],X).
tverb(find, [find |X],X).
tverb(go, [go |X],X).
tverb(have, [has |X],X).
tverb(have, [have |X],X).
tverb(hear, [hear |X],X).
tverb(bump, [bump |X],X).
tverb(handle, [handle |X],X).
tverb(hog, [hog |X],X).
tverb(hold, [hold |X],X).
tverb(improve, [improve |X],X).
tverb(inform, [inform |X],X).
tverb(know, [know|X],X).
tverb(like, [like|X],X).
tverb(limit, [limits |X],X).
```

```
tverb(mean, [mean|X],X).
tverb(mean, [means|X],X).
tverb(need, [need |X],X).
tverb(nominate, [nominate |X],X).
tverb(notify, [notify |X],X).
tverb(offer, [offered |X],X).
tverb(offer, [offering |X],X).
tverb(open,[open|X],X).
tverb(pause, [pause |X],X).
tverb(press, [press|X],X).
tverb(provide, [provide|X],X).
tverb(read, [read|X],X).
tverb(read, [reading|X],X).
tverb(reply, [reply|X],X).
tverb(require, [required|X],X).
tverb(require, [requires|X],X).
tverb(return, [return|X],X).
tverb(satisfy, [satisfies|X],X).
tverb(select, [select |X],X).
tverb(send, [send |X],X).
tverb(set, [set |X],X).
tverb(speak, [speak |X],X).
tverb(specify, [specify|X],X).
tverb(stream, [streaming |X],X).
tverb(makesure, [makesure |X],X).
tverb(take, [take |X],X).
tverb(tap, [tap |X],X).
tverb(turn, [turn |X],X).
tverb(use, [use |X],X).
tverb(use, [uses |X],X).
tverb(want, [want |X],X).
tverb(watch, [watch |X],X).
tverb(watch, [watching |X],X).

relative(who,[who|X],X).
relative(which,[which|X],X).
```

```
relative(whose,[whose|X],X).
relative(that,[that|X],X).
```

Appendix 2.

Sentence list (Extracted)

EN source	Baseline	Reference	EN Improved	Improved	Postedited	CFG results (EN Improved)	Layer Level
Access items that are low in the scan order.	スキャン順の低いアイテムにアクセスします。	スキャンの優先順位が低いアイテムにアクセスします。	We access to the item with low scan priority.	私たちは、低いスキャン優先度でアイテムにアクセスします。	スキャン優先順位が低いアイテムにアクセスします。	s(np(n(we)), vp(tv(access), pp(prep(to)), npp(np(det(the), n(item)), pp(prep(with), np(adj(low), n(scan), n(priority)))))	5
Apps on the device will be available	デバイス上のアプリは利用可能になります	デバイス上のアプリにはアクセス可能です	We can access to apps on the device.	デバイス上のアプリにアクセスできます。	デバイスのアプリにアクセスできるようになります。	s(np(n(we)), vp_aux(aux(can), vp(vp(tv(access), pp(prep(to)), npp(np(n(app)), pp(prep(on), np(det(the),	4

						n(device)))))))	
But it will require the customer to login, which will automatically fail.	しかし、ログインを要求し、自動的に失敗します。	ただし、ログインが必要になるため、自動的に失敗します。	It automatically fails because it requires the customer to login.	顧客がログインする必要があるため、自動的に失敗します。	お客様がログインする必要があるので、自動的に失敗します。	s(npr(np(n(it)) , verb(vpr(vp_adv (adv(automatically), iv(fail)), conj(np(n(it)))))), vp(tv(require), npp(np(det(the) , n(customer)), pp(prep(to), np(n(login))))) ;	5
Your team will be available to help you identify what is controllable	あなたのチームは、制御可能なものを特定します。	チームは、何がコントロール可能かを特定します。	The team will specify the things which are controllable.	チームは、制御可能なものを指定します。	チームがコントロール可能なものを特定します。	s(np(det(the), n(team)), vp_aux(aux(can) , vp(vp(tv(specify), npr(np(det(the) , n(thing)),	5

						rel(vp_bv(bv(be), adj(controllable)))))))	
Customer is comfortable in English.	お客様は英語で快適です。	英語でのやり取りについて、お客様が抵抗を示さないこと。	Customer feels comfortable with English conversation.	お客様は英会話に快適です。	お客様は、英会話に問題がありません。	s(np(n(customer)), vp(tv(feel), npp(np(adj(comfortable)), pp(prep(with), np(n(english), n(conversation)))))	5
Experiencing apps that are freezing to open?	フリーズ、開くことを拒否しているアプリを経験していますか？	アプリがフリーズする、または開かない場合	The apps are freezing to open.	アプリが開くためにフリーズしています。	アプリがフリーズして開きません。	s(np(det(the), n(app)), vp_aux(aux(be), vp(vp(iv(freeze), pp(prep(to)), vp(tv(open)))))	3
The device come into contact with another object.	デバイスは別のオブジェクトと接触します。	デバイスをある物体にぶつけたと考えられます。	You should bump the device against the object.	デバイスをオブジェクトにぶつける必要があります。	デバイスを物体にぶつけたと考えられます。	s(np(n(you)), vp_aux(aux(should), vp(vp(tv(bump), npp(np(det(the)	5

						, n(device)), pp(prep(against), np(det(the), n(object)))))))	
It might have caused the issue.	問題の原因になっている可能性があります。	それが原因です。	It is a cause.	それは原因です。	それが原因です。	s(np(n(it)), vp_bv(bv(be), np(det(a), n(cause))))	3
Having one stuck causes the device to automatically go into Safe Mode.	1 つが詰まっていると、デバイスが自動的にセーフモードになります。	ボタンが正常に上下しない場合、デバイスが自動的にセーフモードになってしまいます。	The device will go into Safe Mode automatically if one button is stuck.	ボタンが 1 つ詰まっていると、デバイスは自動的にセーフモードになります。	あるボタンが動かないと、デバイスは自動的にセーフモードになります。	s(npr(np(det(the), n(device)), verb(vp_aux(aux(will), vp(vp(tv(go), pp(prep(into), np(adj(safe))), npr(np(n(mode), adv(automatically)), conj(np(adj(one), n(button))))))), vp_bv(bv(be), adj(stuck)))	4

I am sorry.	ごめんなさい	残念ですが	I am sorry.	申し訳ありません。	申し訳ありません。	s(np(n(i)), vp_bv(bv(be), adj(sorry)))	3
the device is marked as lost or stolen.	デバイスが紛失または盗難のマークが付けられています。	デバイスは紛失されたものと報告されています。	The device is reported as lost.	デバイスが紛失したと報告される。	デバイスは紛失されたものと報告されています。	s(np(det(the), n(device)), vp_bv(bv(be), ppl(report), pp(prep(as), np(adj(lost)))))	3
As representatives of the corporate, our tolerance of these matters is not only expected but required.	企業の代表として、これらの事項に対する当社の許容性は期待されるだけでなく、必要となります。	企業の代表として、このような問題に辛抱強く取り組まなければなりません。	We must address to these matters patiently as corporate representatives.	これらの事柄には、企業の代表者として辛抱強く取り組む必要があります。	企業の代表者として、これらの事柄に辛抱強く取り組んでいく必要があります。	s(np(n(we)), vp_aux(aux(must), vp(vp(tv(address), pp(prep(to), np(n(these))), npp(np(n(matters), n(patiently)), pp(prep(as), np(n(corporate), n(representatives))))))))	5

Perform a secondary action on an item.	アイテムに対してセカンダリアクションを実行します。	アイテムに対して、2 番目のアクションを実行します。	The secondary action is performed on an item.	第 2 のアクションがアイテムに対して実行されます。	アイテムに対して、2 番目のアクションが実行されます。	s(np(n(the), n(secondary), n(action)), vp_bv(bv(be), ppl(perform), pp(prep(on), np(n(an), n(item)))))	4
Presses the button will be ignored within this number of seconds.	この秒数以内に、ボタンを押しても無視されます。	この秒数以内にボタンを押しても、無視されます。	It is ignored if you press the button within the specified seconds.	指定した秒以内にボタンを押すと、無視されます。	指定した秒数以内でボタンを押すと、無視されます。	s(npr(np(n(it)), verb(vpr(vp_bv(bv(be), ppl(ignore)), conj(np(n(you)))))), vp(tv(press), npp(np(n(the), n(button)), pp(prep(within), np(n(the), n(specified), n(seconds))))))	6

Since there is a conversion involved, some users might not like the colors on their TVs.	変換が関係しているので、一部のユーザーはテレビの色が気に入らないかもしれません。	色変換処理が行われているため、ユーザーによっては、TV の色合いが気に入らない可能性もあります。	Some users might not like the colors on their television because color conversion is performed.	色の変換が実行されるため、テレビの色が気に入らないユーザーもいます。	色変換処理が行われているため、テレビの色が気に入らないユーザーもいます。	s(npr(np(adj(some), n(users)), verb(vpr(vp_aux(aux(mightnot), vp(vp(tv(like), npp(np(det(the), n(color)), pp(prep(on), np(n(their), n(television))))), conj(np(n(color), n(conversion))))), vp_bv(bv(be), ppl(perform)))	5
The Account Owner can nominate someone in the family to be the new Account Owner.	アカウント所有者は、ファミリー内の誰かを新しいアカウント所有者に指名することができます。	アカウント所有者は、メンバーの誰かをアカウント所有者として指名できます。	The Account Owner of the folder can nominate someone of the member as the owner.	フォルダのアカウント所有者は、メンバーの誰かを所有者として指名できます。	フォルダのアカウント所有者は、メンバーの誰かを所有者として指名できます。	s(npp(np(det(the), n(account), n(owner)), pp(prep(of), np(det(the), n(folder)))), vp_aux(aux(can), vp(vp(tv(nomina	7

						te), npp(np(n(someone)), pp(prep(of), np(det(the), n(member)), pp(prep(as), np(det(the), n(owner))))))))	
The user should select the input where the device they are adding connects to.	ユーザーは、追加するデバイスが接続する入力を選択する必要があります。	ユーザーは、追加するデバイスを接続する入力を選択する必要があります。	The user should select the input to connect the additional device.	ユーザーは、追加のデバイスを接続するための入力を選択する必要があります。	ユーザーは、追加デバイスを接続する入力を選択する必要があります。	s(np(det(the), n(user)), vp_aux(aux(should), vp(vp(tv(select), npp(np(det(the), n(input)), pp(prep(to), vp(tv(connect), np(det(the), adj(additional), n(device)))))))	4

Use a 3-finger tap and hold to the end of the selection.	3 本指でタップし、選択範囲の最後まで押し続けます。	選択範囲の最後になったら、もう一度 3 本の指でタップして長押します。	You must tap and hold with three fingers to the end of the selection.	選択の最後へ 3 本の指でタップして保持する必要があります。	選択範囲の最後の部分まで、3 本の指でタップして長押しする必要があります。	s(np(n(you)), vp_aux(aux(must), vp(vp(tv(tap), npr(np(n(and)), verb(vp(tv(hold), pp(prep(with), np(adj(three))), npp(np(n(finger)), pp(prep(to), np(det(the), n(end)), pp(prep(of), np(det(the), n(selection))))))))))	8
Users will be able to see the missed alerts in the Notifications Center, when turning their TV on.	ユーザーは、テレビの電源を入れたときに、通知センターで不在通知を見ることができます。	見逃したアラートについては、TV をオンにすると、通知センターで確認できます。	You can check the missed alerts in the Notification Center when you turn on television.	テレビの電源を入れたときに、通知センターで不在通知を確認できます。	テレビの電源を入れたときに、見逃したアラートを通知センターで確認できます。	s(npr(np(n(you)), verb(vpr(vp_aux(aux(can), vp(vp(tv(check), npp(np(det(the)	5

						, adj(missed), n(alert)), pp(prep(in), np(det(the), n(notification), n(center))))))), conj(np(n(you))))), vp(tv(turn), pp(prep(on)), np(n(television))))	
The device dropped.	デバイスがドロップされました。	デバイスを落としました。	You dropped the device.	デバイスを落としました。	デバイスを落としました。	X = s(np(n(you)), vp(tv(drop), np(det(the), n(device))))	3
Without lifting your finger, draw circles clockwise.	指を持ち上げることなく、時計回りに円を描きます。	指を持ち上げずに、時計回りに円を描くと	You must draw circles clockwise without lifting your finger.	指を持ち上げずに時計回りに円を描く必要があります。	指を持ち上げずに、時計回りに円を描く必要があります。	s(np(n(you)), vp_aux(aux(must), vp(vp(tv(draw), npp(np(n(circle), adv(clockwise))	5

						, pp(prep(without), np(n(lifting), n(your), n(finger)))))))	
You must be an adult to hold an account.	アカウントを保持するには大人でなければなりません。	サービスを使用するためには、アカウントが必要で	An account is required for service.	サービスにはアカウントが必要です。	サービスにはアカウントが必要です。	s(np(det(a), n(account)), vp_bv(bv(be), ppl(require), pp(prep(for), np(n(service)))))	4
You also need an account to use the service.	また、サービスを使用するにはアカウントが必要です。	アカウントを所持するには成人である必要があります。	You must be an adult to hold an account.	アカウントを保持するには、大人である必要があります。	アカウントを保持するには、成人である必要があります。	s(np(n(you)), vp_aux(aux(must), vp(vp_bv(bv(be), npp(np(det(a), n(adult)), pp(prep(to), vp(tv(hold), np(det(a), n(account))))))))	5

ADD IF USER HAD A POOR EXPERIENCE:	ユーザーが貧弱な経験を持っていた場合は追加します。	ユーザーが不快な思いをした場合に追加：	You can add it if the user was uncomfortable.	ユーザーが不快だったら追加してもいいよ	ユーザーが不快に感じられた場合、これを追加できます。	s(npr(np(n(you)), verb(vp_aux(aux(can), vp(vp(tv(add), npr(np(n(it)), conj(np(det(the), n(user))))))))), vp_bv(bv(be), np(adj(uncomfortable))))	4
You have every right to be upset.	あなたは動揺するあらゆる権利を持っています。	お客様がご立腹になるのは当然のことです。	It is natural that you are upset.	あなたが動揺するのは当然です。	あなたが動揺するのは当然です。	s(npr(np(n(it)), verb(vp_bv(bv(be), npr(np(adj(natural)), rel(np(n(you))))))), vp_bv(bv(be), np(adj(upset))))	4