

Extracting and Using Translation Templates in an Example-Based Machine Translation System

Ethel Ong¹, Kathleen Go¹, Manimin Morga¹, Vince Nunez¹, and Francis Veto¹

College of Computer Studies

De La Salle University-Manila

¹onged@dlsu.edu.ph

A bidirectional English-Filipino machine translation system is developed that extracts translation templates and chunks from a given bilingual English-Filipino corpus. These templates and chunks are then used to translate an input English document to Filipino and vice versa. The system extended the similarity and difference translation template learning algorithms of Cicekli and Guvenir (2003) by refining existing templates and deriving templates from previously learned chunks. Chunk alignment, splitting algorithms, and chunk refinement are also introduced in the training process. Correct extraction of similarity templates and chunks during the learning process led to translation with a low word error rate of 15% for a test document whose sentences exactly match the training set, to a high 86% when the test document is different from the training corpus. Using difference templates alone, the resulting translation has a word error rate of 49% to 85%. Combined use of similarity and difference templates resulted in a low word error rate of 18% when the test document contains sentence patterns matching the training set, to a high 85% when the test document is different from the training corpus. Tests also showed that the translation with the highest score selected from a set of candidate translations is consistently the best choice as validated against automatic evaluation methods.

Keywords: Template Learning, Example-Based Machine Translation, Bilingual Corpora

INTRODUCTION

Computer-based machine translation systems provide a means for people to exchange information more efficiently by allowing individuals to translate a body of text from one language to another. Translation between natural languages has been a major area of study in natural language processing. Various machine translation systems have been constructed that use two primary approaches, namely rule-based machine translation (RBMT) which makes use of rules of the language, and

example-based machine translation (EBMT) which relies on the information fed into the system, most of which are taken from aligned bilingual corpora.

RBMT systems parse documents and use a symbolic representation to generate the translation output afterwards. They require a large set of rules and extensive lexicons with morphologic, syntactic, and semantic information to perform their task. Furthermore, these systems generate more and more rules which often lead to more complexity and sophistication, making them quite difficult to maintain (Hovy, et al., 2001).

EBMT systems learn how a certain sentence is to be translated by being trained on a given bilingual corpus which contains a set of sentences in the source language with a corresponding translation in the target language. Correspondences between the sentences are learned by the system and subsequently used in translation. Translations are generated by comparing the input from a source language to the examples or templates stored in a database, finding the best match possible before deducing the equivalent text in the target language.

TExt Translation is an EBMT system that translates English sentences to Filipino and vice versa. This approach is used to maximize the amount of information derived from the input corpora, specifically example translation sentence pairs between source and target languages using the least amount of linguistic resources. Utilizing templates is more flexible as compared to word-for-word translation (Kaji, 1992) and to pre-defined rules which would be hard to modify at a later time. Moreover, as the Filipino language is evolving in response to calls for globalization, chunks learned during training can supplement limited entries in the lexicon and are used in further learning sessions to extract more templates from the corpus.

TRANSLATION TEMPLATES AND CHUNKS

A translation template is defined by Kaji (1992) as “a bilingual pair of sentences in which corresponding units (words and phrases) are coupled and replaced with variables”. Kinoshita et al., (1994) defines a translation template to contain a pair of source and target patterns that consist of constants and variables, where a source pattern is used as a comparison reference to the sentence being translated and the target pattern is used to generate the translation of the input. The patterns preserve the ordering of words in the translation, regardless of the variance in the sentence structures of the source and target languages.

New templates can be learned from the given bilingual corpus by using a heuristic called

Translation Template Learner (TTL), which is presented in Cicekli and Güvenir, (2003). TTL analyzes the similarities and differences between a pair of translation example sentences. A similarity template is a non-empty sequence of common items in both sentences and is learned by preserving the similar parts in the sentences, and replacing the differences with variables to form a translation template between two sentences of a language. A difference template is a pair of two sequences (**D1**, **D2**) where **D1** is the sub-sequence of the first sentence while **D2** is the sub-sequence of the second sentence, and **D1** and **D2** do not contain a common item.

Kaji (1992) presented two steps in acquiring translation templates. First, paired sentences from the source and target languages are coupled by analyzing parts of the inputs and pairing when equivalences (matching word translations) are found in the dictionary (lexicon). The paired units are then selected and substituted with unique variables. This process may result in creating a sentence template, or fragments of a sentence (called chunks in the TExt system). The second step involves refining the learned templates to discard useless ones.

Given the example sentence pairs **S1** and **S2**:

S1: The dog ran. ↔ Tumakbo ang aso.

S2: The cat ran. ↔ Tumakbo ang pusa.

One similarity template (**T1**) and two difference templates (**T2** and **T3**) will be learned:

T1: The [1] ran. ↔ Tumakbo ang [1].

T2: [2] dog [3] ↔ [3] [2] aso.

T3: [2] cat [3] ↔ [3] [2] pusa.

Using the lexicon to align the corresponding English and Filipino words in the input sentences, the system retains the tokens “**The**”, “**ran**”, and “**Tumakbo ang**” as the constants of similarity template **T1**, while “**dog**”, “**aso**”, “**cat**” and “**pusa**” are retained as the constants of difference templates **T2** and **T3**, respectively. [1], [2],

and [3] are variables in the template and are referred to as atomic templates (Cicekli and Güvenir, 2003). In TExt, an atomic template is called a chunk, which is a sequence of words used to substitute the variables in a given template. TExt-2, an extension to TExt, performs chunk refinement to generalize newly acquired chunks with previously learned chunks. Chunk refinement is discussed further in Section 4.3.

Chunks have similar representation and usage as templates. They both represent the correspondence between the source and target languages. It is not possible for a template to have no chunks. While it is possible for chunks to contain other chunks, it is not a requirement. A chunk may contain any number of words or phrases, and may also contain other chunks. The difference between a chunk and a template is that templates are patterns of complete sentences while chunks are not.

In TExt, chunks in templates are represented by a numeric value, e.g., [1]. This is referred to as the domain of the chunk. The domain allows the chunks to have a reference from the template from which they came from. Specific chunks are labeled following the format [X.n], where X represents the domain of the chunk, and n represents the sequence number of the chunk within the domain. Only the domain is important to identify if a chunk is an exact match for the template during translation. For example, if the domain in the template is [X], then any chunk with a domain “X” is an exact match for the template and can be used during translation.

Using the given sentence pairs S1 and S2 as examples, chunks [1.1] and [1.2] will be learned by TExt. If TExt learns another chunk [1.3] from a different set of input sentence pairs in a later training session, then all these chunks can be used during translation to substitute the variable [1] in the similarity template T1:

[1.1]: dog ↔ aso
 [1.2]: cat ↔ pusa
 [1.3]: children ↔ mga bata

For templates T2 and T3, chunks [2.1] and [3.1] can be learned from the input sentence pairs S1 and S2 and used to substitute the variables [2] and [3], respectively, during translation. Another chunk [3.2] (again, possibly learned from a different set of input sentence pairs), can also be used to replace variable [3]:

[2.1]: The ↔ ang
 [3.1]: ran ↔ tumakbo
 [3.2]: is eating ↔ kumakain

The rest of this document discusses the architectural design, learning and translation algorithms, and test results of TExt and TExt-2. From here onwards, TExt will be referred to as TExt-1, to distinguish it from its successor, TExt-2. Section 3 presents the architecture of both TExt-1 and TExt2, as well as the similarity template learning and refinement algorithms of TExt-1. Section 4 presents TExt-2 and its algorithms for learning difference templates and chunk refinement. Section 5 provides the results gathered from training TExt-1 on four sets of corpus, as well as the translation quality when utilizing these learned templates. Comparisons are also provided to determine the training and translation performance of the system when learning and using similarity templates alone (TExt-1), difference templates alone, and combined similarity and difference templates (TExt-2).

THE TEXT TRANSLATION SYSTEM

Text Translation (TExt-1 and TExt-2) has four main components, as shown in Figure 1. These are input analysis, template extraction during training, translation, and knowledge sources. During input analysis, the bilingual corpus is processed following three main steps: sentence segmentation, tokenization, and unit alignment. Sentence segmentation divides the input text into sentences. Tokenization then divides the sentences into smaller fragments such as words, punctuation marks and other symbols. Unit alignment aligns corresponding tokens between the source sentence and the target

sentence according to the entries in the lexicon. The output of the analysis module is passed to the training module or the translation module.

In the training module, aligned sentence units are analyzed to acquire templates and chunks. Three approaches were employed to extract templates from the input; these are template refinement, deriving templates from sentences, and deriving templates from chunks. During translation, input sentence tokens are analyzed to collect candidate templates and chunks from the knowledge base that can be used to generate the output sentence in the target language.

The training and translation modules utilize a number of linguistic resources, namely, the lexicon, morphological lookup table, templates database, chunks database, unused sentences repository, and common and noise words tables.

The lexicon currently stores 10,500 English and Filipino word pairs, and is used in the input analysis and translation phases to identify matching sentence

tokens. The morphological lookup table is patterned after TWIRL (Ang, et al., 2005), an English-to-Filipino machine translation system. Currently containing 10,200 entries of English words and 3,000 entries of Filipino words, it is used as a reference for root words of sentences and forms of the morphologically modified words.

The templates database and the chunks database are initially empty. As training progresses, these two databases are used to store all templates and chunks learned from the given corpus. During translation, these databases are accessed to locate candidate templates and chunks that can be used to generate the output sentence in the target language. Templates are stored as pairs consisting of an English template and the corresponding Filipino template with variables referring to matches in the Chunk Database. Stored templates can be used in bidirectional training and translation – from English to Filipino or vice versa – and can have one-to-many template correspondences

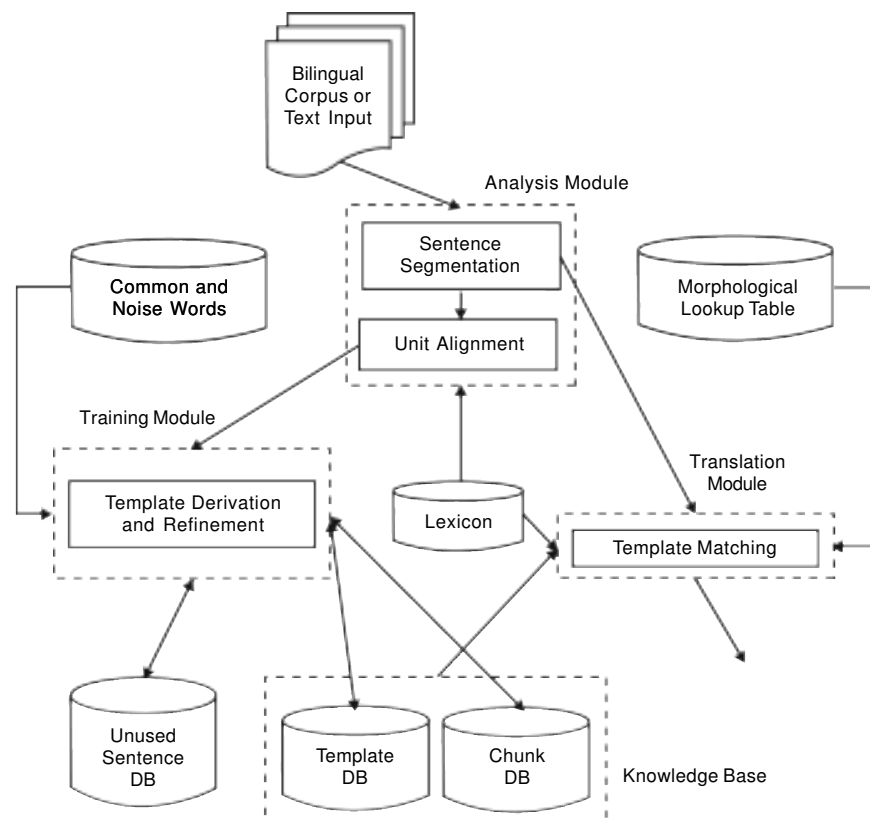


Figure 1. Architecture of TText Translation

between languages – one English template can have multiple equivalent Filipino templates.

The unused sentences repository stores all previously learned sentences for which no templates have been constructed yet. These sentences can be used in subsequent training sessions, when additional corpus is fed to the system. Finally, the common and noise words tables are used as references to filter common words frequently found in usage in English and Filipino and noise words in either the source or target language which do not necessarily have matches in the other.

Template Refinement

Template Refinement (TR) compares an aligned sentence pair against existing templates in the templates DB. An aligned sentence is said to match a given template whenever it contains a token that matches exactly with a corresponding token in the template itself. There must be a corresponding match in both the source and target languages in order for the template to be considered. Through these similarities a candidate refinement is identified. Consider the input sentence pair **S1** and an existing template **T1**:

S1: I ate meat ↔ **Kumain ako ng karne**

T1: I ate [1] ↔ **Kumain ako ng [1]**

TR matches **S1** to **T1** because they both contain “**I ate**” and “**Kumain ako ng**” in their source and target sentences, respectively. The identified differences, “**meat**” and “**karne**”, are mapped as a new chunk in chunk domain [1]. Assuming that this chunk domain [1] already contains chunk [1.1] that has been acquired in a previous training session, the next chunk label available is used:

[1.1]: **fish** ↔ **isda**

[1.2]: **meat** ↔ **karne**

There are cases when not all constants in a given template match those of an aligned sentence pair. Consider the input sentence pair **S2** and an existing template **T2**:

S2: We calmly went to the hospital.

↔ **Mahinahon kaming pumunta sa pagamutan.**

T2: We happily went to [2] ↔

Masaya kaming pumunta sa [2]

Only the underlined segments match between **S2** and **T2**. TR does not make a new template; instead it refines **T2** to reflect its remaining new similarities with **S2**. Template **T2** is discarded; a new template, **T3**, and a new chunk domain [3] are created:

T3: We [3] went to [2] ↔

[3] kaming pumunta sa [2]

[2.5]: the hospital ↔ **pagamutan**

[3.1]: happily ↔ **masayang**

[3.2]: calmly ↔ **mahinahong**

Derivation of Templates from Sentences

If an aligned sentence pair does not have a token matching those of any of the templates in the knowledge base, the sentence pair is then compared with other aligned sentence pairs. This process is termed Derivation of Templates from Sentences (DTS). Consider the input sentence pairs **S3** and **S4**:

S3: The woman walked happily. ↔

Masayang naglakad ang babae.

S4: The couple walked happily. ↔

Masayang naglakad ang mag-asawa.

The DTS algorithm takes all similar tokens between the source and target sentences in **S3** and **S4** (underlined) and preserves them as constants in the new template **T4** while the differing elements are created as chunks in [4].

T4: The [4] walked happily ↔

Masayang naglakad ang [4]

[4.1]: woman ↔ **babae**

[4.2]: couple ↔ **mag-asawa**

There are cases when a single long differing chunk in the source sentences corresponds to two short differences in the target sentences. Using **S3** once more and a new sentence **S5**:

S3: The woman walked happily. ↔
Masayang naglakad ang babae.
S5: The man watched happily. ↔
Masayang nanood ang lalaki.

Upon initial inspection, the source sentences have a single long differing chunk, “**woman walked**” for **S3** and “**man watched**” for **S5**, whereas in the target sentences there are two differing simple chunks, namely, “**naglakad**” and “**babae**” for **S3** and “**nanood**” and “**lalaki**” for **S5**. In this case, the single long chunk is split into two shorter chunks to see whether the resulting subchunks would match either of the corresponding differing chunks in the target. This process is called Strict Chunk Alignment with Splitting (SCAS), and it generates the template **T5** and chunk domains [5] and [6]:

T5: The [5] [6] happily ↔
Masayang [6] ang [5]
[5.1]: woman ↔ **babae**
[5.2]: man ↔ **lalaki**
[6.1]: walked ↔ **naglakad**
[6.2]: watched ↔ **nanood**

Derivation of Templates from Chunks

When templates cannot be derived from aligned sentence pairs using TR and DTS, the system then performs Derivation of Template from Chunks (DTC). While the system is learning templates from aligned sentence pairs, it is able to simultaneously extract a significant number of chunks. As there are no other resources available other than the lexicon which could be used in the learning process, the chunks are also reused because they are additional information from actual examples. Consider the new sentence **S6** and existing chunks from the knowledge base:

S6: Filipinos are cheerful and hospitable. ↔ **Masayahin at mapanauhin ang mga Pilipino.**
[6.4]: Filipinos ↔ **mga Pilipino**
[7.9]: hospitable ↔ **mapanauhin**

DTC simply takes matching chunks from the knowledge base (chunks [6.4] and [7.9]) and use them to substitute parts of an aligned sentence. In this example, **S6** is transformed to generate the new template **T6**:

T6: [6.4] are cheerful and [7.9]
 ↔ **Masayahin at [7.9] ang [6.4]**

Translating Sentences Using Templates

The templates learned during training can be directly used to perform bi-directional translation. During translation, input sentence tokens are analyzed to collect candidate templates and chunks from the knowledge base. A template or a chunk is considered a candidate if it has at least one word used in the input sentence. Consider the input sentence **S7**:

S7: The pretty girl ran.

All chunks and templates having the words “**The**”, “**pretty**”, “**girl**”, and “**ran**” will be added to the list of candidates. This is necessary because any word in the input sentence can be a variable and it is possible that only one constant (or word) is present in the candidate.

The list of candidates are then reviewed and filtered to exclude inappropriate matches. A candidate is an inappropriate match if it contains words that are not found in the input sentence, if it is longer than the sentence, or if the word order is different from the input sentence. For example, only the chunk “**pretty [28]**” will be useful for input sentence **S7**. Chunks “**boy and girl**” and “**ran away**” are inappropriate and will be removed from the candidates list.

The remaining candidates are assigned scores according to the structure of the template or chunk,

the presence or absence of chunk variables in templates, and the presence of word matches in templates. The translation output that produces the highest total score is used for the given input. If there are candidates having the same score as the highest scorer, the first candidate with the highest score will be selected.

Given the input sentence **S8** and candidates **T12** and **[8.1]**:

S8: I ate vegetables.
T12: [8] ate [1].
[8.1]: I

In the first iteration of the scoring algorithm, **T12** will receive points for the matching string constant “ate” and for the presence of the two chunk variables **[8]** and **[1]**. During the second iteration, **[8]** will be replaced with **[8.1]** resulting in the partial match “**I ate [1].**” Additional points will be added due to the matching word “**I**”, and because the assigned chunk **[8.1]** has the same domain as the target chunk **[8]**. However, for the chunk **[1]**, no match can be found. No points will be given for this chunk since word-for-word translation will be performed.

Note that during the candidate identification process, the matching chunks may have domains that are different from the one specified in the matching template. This allows the use of chunks from various domains as candidates in lieu of the lexicon which uses word-for-word translation. However, in the scoring process, a match that has the same domain as specified will be given a higher score.

During learning, the extracted templates and chunks have bi-directional properties; the same set of templates and chunks will be learned for a given training set, regardless if the source language is English or Filipino. This same property should hold true during translation. If an exact match is formed for a given sentence in English, the same candidate would be formed for the Filipino sentence. However, it is possible that a template or chunk in the source language contains no constants but the one in the target language does. The bi-directional

property will not hold in this situation, and the resulting translations in both directions can differ.

LEARNING DIFFERENCE TEMPLATES

Text-2 extended the learning process of TExt-1 with the integration of a difference template learning algorithm and the implementation of chunk refinement. Both similarity and difference templates are stored in the same templates database. This allows the translation algorithm to consider both types of templates as candidates for a given input sentence to be translated, following the selection and matching criteria of TExt-1.

Modified Derivation of Templates from Sentences (DTS)

The modified DTS algorithm starts by determining the similarities and differences in the input sentence pairs. Consider the input sentence pairs **S21** and **S22**:

S21: My favorite pet is a dog. ↔ Aso ang aking paboritong alaga.
S22: My favorite color is red. ↔ Pula ang aking paboritong kulay.

All similar tokens between **S21** and **S22** (underlined) are preserved as constants in the new similarity template **T21** while the differing elements are created as chunks **[21]** and **[22]**:

T21: My favorite [21] is [22] ↔ [22] ang aking paboritong [21]
[21.1]: pet ↔ alaga
[21.2]: color ↔ kulay
[22.1]: a dog ↔ aso
[22.2]: red ↔ pula

All differing tokens are preserved as constants in the new difference templates **T22** and **T23** while the similar element is created as a new chunk **[23]**:

T22: [23] pet is a dog ↔
Aso ang [23] alaga.

T23: [23] color is red ↔
Pula ang [23] kulay.

[23.1]: **My favorite** ↔ aking paboritong

Note that alignment is a factor in determining the similarities and differences. If a word in the source language is not aligned with a word in the target language, it will not be included in the list of similarities and differences. This is because if it is permitted, there would be unequal number of chunks between the source and target language templates. During unit alignment, not all words in English and in Filipino have corresponding words in the other language. Words that are not aligned are retained as string constants in the templates, for example, in **T22** and **T23**, these are the underlined words “**is**” and “**ang**”. The sentence containing words without alignments can still be used if other words in the sentence have alignments.

Effect of Learning Difference Templates to Deriving Templates from Chunks (DTC)

The difference template learning (DTL) algorithm always generates two difference templates for each set of matching input sentence pairs. Consider the input sentence pairs **S23** and **S24**:

S23: The pretty Sampaguita is the national flower. ↔
Ang marikit na Sampaguita ay ang pambansang bulaklak.

S24: The fierce Philippine Eagle is the national bird. ↔
Ang mabangis na Philippine Eagle ay ang pambansang ibon.

During DTS, similarity template **T24**, difference templates **T25** and **T26**, as well as chunks [24], [25], [26], and [27] would be learned:

T24: the [24] [25] is the national [26] ↔ ang [24] na [25] ay ang pambansang [26]

T25: [27] pretty Sampaguita is the national flower ↔ [27] marikit na Sampaguita ay ang pambansang bulaklak

T26: [27] fierce Philippine Eagle is the national bird ↔ [27] mabangis na Philippine Eagle ay ang pambansang ibon.

[24.1] pretty ↔ marikit

[24.2] fierce ↔ mabangis

[26.1] flower ↔ bulaklak

[26.2] bird ↔ ibon

[27.1] the ↔ ang

Note that chunk [27] “the ↔ ang” was learned because of DTL. When a new input sentence pair **S25** is encountered, both TR and the modified DTS cannot be used to derive a difference template due to the absence of a matching sentence pair. DTC will be called instead to derive template **T27** from the existing chunk [27]:

S25: The whaleshark, or the biggest fish in the world, is found here. ↔ Ang butanding, o ang pinakamalaking isda sa mundo, ay matatagpuan dito.

T27: [27] whaleshark, or the biggest fish in the world, is found here. ↔ [27] butanding, o ang pinakamalaking isda sa mundo, ay matatagpuan dito

This example shows that if an input sentence pair has no match, DTL will still generate more templates than the similarity template learning (STL) algorithm because DTL resulted in more frequent calls to the DTC process of TExt-1 to derive new templates from existing chunks.

Chunk Refinement

TExt-2 performs Chunk Refinement (CR) prior to storing a newly acquired chunk into the database. CR compares the new chunk with an existing chunk for alignment. A new chunk is said to be aligned to an existing chunk if the new chunk contains a token that matches exactly with a corresponding token in the existing chunk. There must be a corresponding match in both the source and target languages in order for the chunk to be considered. Similarities and differences between the aligned chunks are identified as candidates for refinement. New chunks are learned by retaining the similarities as well as the differences between the aligned chunks. Learning from the similarities and differences increases the number of chunks learned and consequently increases the number of chunks which can be used in translation.

Consider an existing chunk [19]:

[19.1] **big cat** ↔ **malaking pusa**

This chunk can only be used if the words to be translated are “**big cat**” or “**malaking pusa**”. If the system extracts a new chunk [X]:

[X]: **big kid** ↔ **malaking bata**

It identifies the existing chunk [19] as candidate for refinement, forming “**big [28] ↔ malaking [28]**”, which replaces chunk [19] and discards the new chunk. Additional chunks [28], [29], [30], and [31] are also learned in the process:

[19.1] **big [28]** ↔ **malaking [28]**
 [28.1] **cat** ↔ **pusa**
 [28.2] **kid** ↔ **bata**
 [29.1] **big** ↔ **malaking**
 [30.1] [29] **cat** ↔ [29] **pusa**
 [31.1] [29] **kid** ↔ [29] **bata**

Notice that chunk [19] was derived by retaining the similarities between the two aligned chunks, while chunks [30] and [31] were

derived by retaining the differences. Generalizing chunks makes them more useful and increases their scope, enabling the system to use these chunks to translate different phrases. In the case of chunk [19], it can now be used to translate phrases with “**big**” or “**malaking**”, for example “**big house**” and “**malaking sanggol**”, while chunk [30] can be used to translate “**fat cat**” and “**lazy cat**”.

TEST RESULTS AND DISCUSSION

Various tests were conducted to determine the quality of both the templates and chunks extracted during training and of the output generated during translation. The presentation of the test results are divided as follows. Sections 5.1 and 5.2 discuss the results of testing the training and translation modules on the two versions of TExt-1’s chunk alignment algorithm and common words filtering algorithm. Sections 5.3 and 5.4 compare the test results of TExt-1 with that of Text-2, highlighting the effects of learning and using similarity templates separate from difference templates, as well as combined similarity and difference templates.

Four sets of corpora were used to train TExt-1; corpus #1 consists of 49 sentence pairs, corpus #2 consists of 15 sentence pairs, corpus #3 consists of 41 sentence pairs, and corpus #4 consists of 58 sentence pairs, totaling 163 sentence pairs. Corpora #1 to #3 were created by the proponents and verified by a linguist. The sentences in these corpora have similar sentence structures so that templates can be learned. Corpus #4 was adapted from a document that was received from the Filipino Department of De La Salle University - Manila. Only those sentences which had the same sentence patterns as the training corpus were selected and included in the translation corpus.

Learning Similarity Templates and Chunks

The Strict Chunk Alignment with Splitting (SCAS) algorithm used in deriving templates from sentences requires that all tokens are aligned and the number of given chunks in the source language

is equal to that in the target. Using this approach, more templates that are of good quality are learned, as shown in Table 1, as compared to the Loose Chunk Alignment approach (LCA). Correctness in Table 1 refers to the actual templates and chunks learned as well as the proper alignment of tokens in the source and target template or chunk.

Further analysis of the extracted templates showed that there are too many frequently

occurring words which do not contribute to the quality of the derived templates. These common words, which were identified based on existing studies of English and Filipino corpora, need to be filtered from becoming the remaining constant in any given template. This is because many aligned sentence pairs would match with such a template, leading to templates with a small coverage during translation.

Table 1. Test Results for Chunk Alignment Algorithms of TExt-1

Corpus Details		
# of sentence pairs in Bilingual Corpus #4		58
	LCA algorithm	SCAs algorithm
English to Filipino		
Total # of template pairs learned	5	13
# of % correct template pairs	3 (60%)	13 (100%)
Total # of chunk pairs learned	110	29
# % correct chunk pairs	49 (44.54%)	29 (100%)
# of unused sentence pairs	6	32
Filipino to English		
Total # of template pairs learned	6	13
# % correct template pairs	2 (33.33%)	13 (100%)
Total # of chunk pairs learned	131	29
% correct chunk pairs	64 (48.85%)	29 (100%)
# of unused sentence pairs	6	32

Tests were also conducted to identify how much more useful the generated templates and chunks would be if a filter for common words was implemented. A version of the training module, called NCWF (No Common Words Filtering), does not implement such a filter while another implementation, called CWF (Common Words Filtering) does. The results in Table 2 indicate that the CWF version produces more templates but fewer chunks.

NCWF generated fewer templates because TR is performed more often to produce templates with common word remainders. This also resulted in having derived more chunks. CWF generated more templates and fewer chunks which is preferable because templates are able to capture proper sentence structures, thus preserving word order in the resulting translation. More templates would also mean more candidates for refinement in subsequent training executions.

Table 2a. Test Results for Filtering Algorithms for Corpora #1-4 (163 sentence pairs)

	NCWF	CWF
Total # of template pairs learned	59	73
Total # of chunk pairs learned	237	210
Total # of unused sentence pairs	29	29

Table 2b. Test Results for Filtering Algorithms for Corpora #4 only (58 sentence pairs). Percentage values show comparison against total numbers in Table 2a.

	NCWF	CWF
# (%) template pairs learned from bilingual corpus #4	31 (52.54%)	30 (41.10%)
# (%) chunk pairs learned from bilingual corpus #4	47 (19.83%)	39 (18.57%)
# (%) unused sentence pairs from bilingual corpus #4	16 (55.17%)	17 (58.62%)

Translating Sentences Using Similarity Templates and Chunks

Automatic evaluation of the translated outputs was carried out using the online MT evaluation tools available in DCU – Dublin City University. Three bilingual corpora, used as input files, were first manually tagged in the Standard Generalized Markup Language (SGML) format required by the automated evaluator. The metrics used are the Word Error Rate, Sentence Error Rate, and Bilingual Evaluation Understudy for the three system versions LCA, SCAS-NCWF and SCAS-CWF. Corpus #3 reflects translations for exactly the same document learned, corpus #5 reflects translations for sentences derived from corpora #1-#4 used for training the system, while corpus #7 reflects translations for sentences derived from

the second half of sentences from corpora #1-#4 of which the first half was used for training.

Word Error Rate (WER) computes the percentage of words which are to be deleted or replaced in the translation with the aim of obtaining the reference sentence. Figure 2 shows that the result of translating from the same corpus produces very low WER (corpus #3). WER is still commendable for translating based on structures from previously learned sentences (corpus #5). Furthermore, translation of sentences which have relatively different structures than those learned lead to low quality output (corpus #7).

The percentage of sentences whose translations have not exactly matched the corresponding sentence in the reference sentences is computed by the Sentence Error Rate (SER). SER takes into consideration the incorrectness of entire sentences. It is therefore understandable that the error rates shown in Figure 3 are somewhat significantly higher than those of WER in Figure 2. The same observations and conclusions regarding the performance of the various versions on the three corpora are the can be made for SER.

The Bilingual Evaluation Understudy (BLEU) metric is a modified n-gram precision measure. It employs a weighted geometric average of n-gram matches between test sentences and reference sentences, which is modified to penalize over-generation of correct word forms. A multiplicative brevity penalty is also incorporated which penalizes test sentences found to be shorter than the reference sentences as computed at the corpus level. The resulting score is a numeric metric intended to indicate the closeness of a set of test sentences to their reference sentences considering the length, word order and word choice. The scoring captures both adequacy and accuracy. Adequacy is satisfied using the same words (unigrams) as in the references. Accuracy is reached with the longer n-gram matches.

BLEU measures the accuracy and adequacy of translations and so it evaluates in a somewhat opposite manner to those for WER and SER, that is, the higher the score, the better. One observation is that even though the exact corpus

used for training is also used for translation, the BLEU scores (Figure 4) are not exactly perfect. This could be attributed to unused sentences for which no templates could be generated and for

which the often unreliable word-for-word translation is used. This case holds true even for the results of WER (Figure 2) and SER (Figure 3).

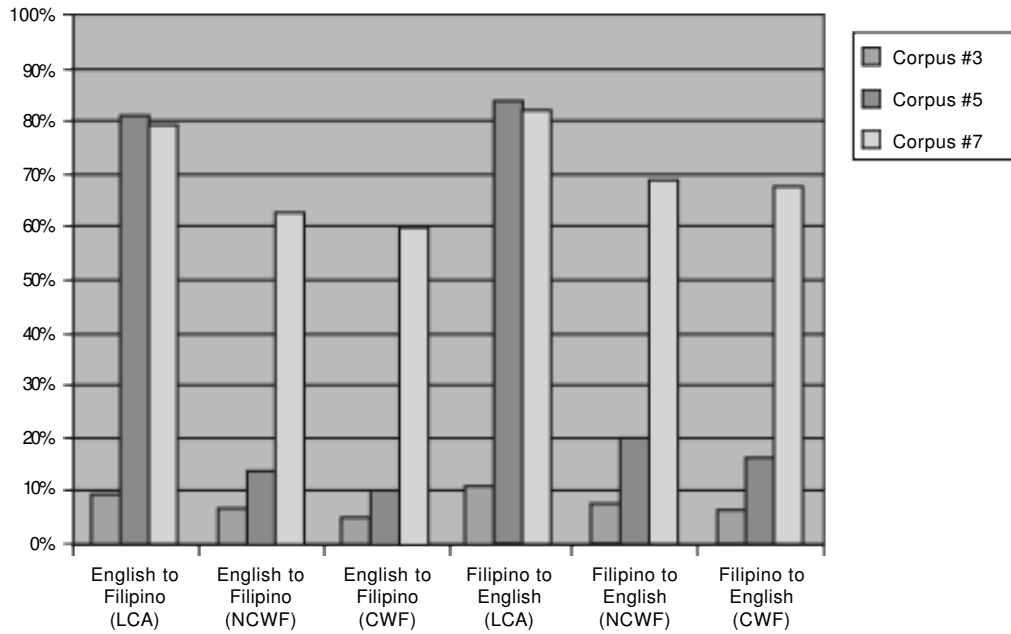


Figure 2. Test Results for Word Error Rate

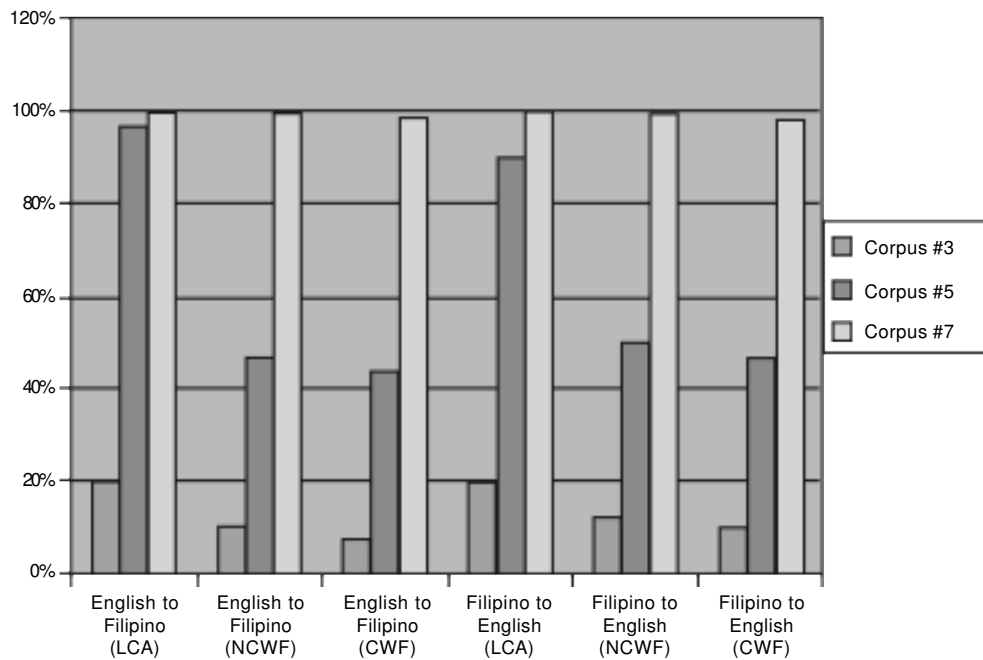


Figure 3. Test Results for Sentence Error Rate

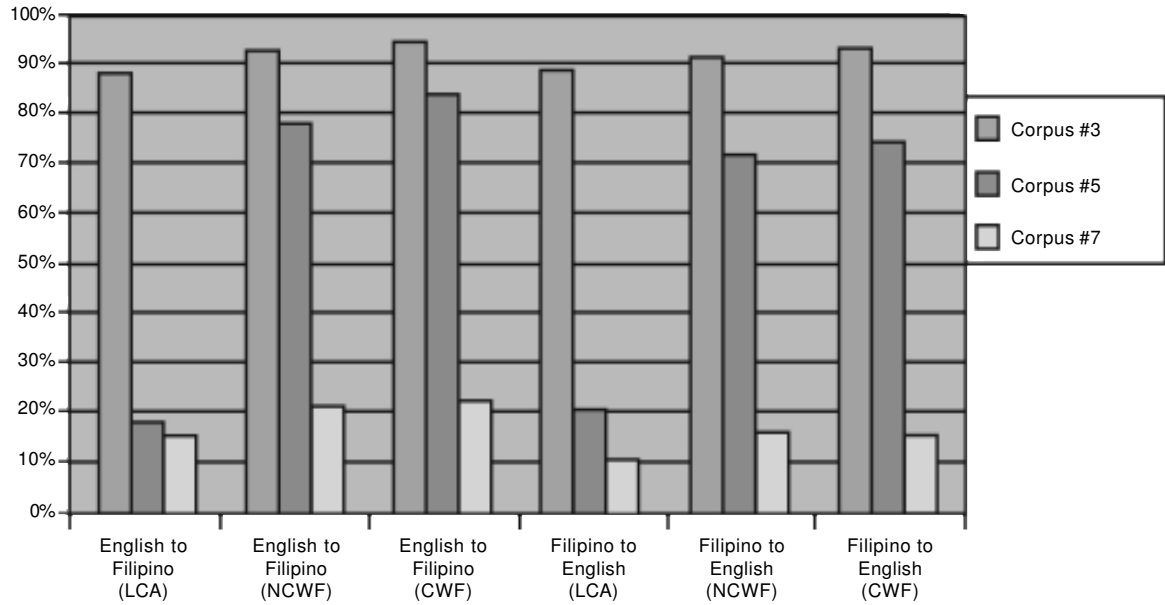


Figure 4. Test Results for BLEU

As can be seen from the test results in Figures 2 to 4, the SCAS-CWF version consistently performs better than the LCA or SCAS-NCWF versions. Correct extraction of templates and chunks that are of good quality during the learning process led to reduced word and sentence error rates by as much as 50% during translation.

Tests were also conducted to determine the correctness of the scoring algorithm. Ten input sentences that were correctly translated by the

system were identified. For each of these sentences, the top five candidate translations are submitted to the automatic evaluator for comparison against the reference translation. Ten input sentences that were incorrectly translated by the system were also evaluated in the same manner. Results presented in Table 3 show that the translation with the highest total score is consistently the best choice as validated against automatic evaluation methods.

Table 3. Test Results for Top 5 Candidates for Correctly and Incorrectly Translated Input Sentences

Correctly Translated				
Top Candidate	Version	WER %	SER %	BLEU %
Top 1	CWF	0.00	0.00	100.00
Top 2	CWF	39.89	70.00	61.29
Top 3	CWF	53.44	100.00	52.26
Top 4	CWF	59.57	100.00	44.73
Top 5	CWF	59.57	100.00	46.96

Incorrectly Translated				
Top Candidate	Version	WER %	SER %	BLEU %
Top 1	CWF	21.73	90.00	58.35
Top 2	CWF	31.03	90.00	51.98
Top 3	CWF	35.99	100.00	41.56
Top 4	CWF	52.81	100.00	27.35
Top 5	CWF	53.72	100.00	27.32

Learning Difference Templates and Chunk Refinement

The same set of corpora (#1-#4) containing 163 sentence pairs used in training TExt-1 was utilized in training TExt-2. Five variants of TExt-2 were tested, to determine the number of similarity and difference templates that are learned separately by each approach and when both approaches are combined, as well as to observe the effects of chunk refinement. The results are summarized in Table 4. Note that the version of TExt-1 uses the strict chunk alignment and common words filtering algorithms. STL refers to Similarity Template Learning approach, DTL refers to Difference Template Learning approach, and CR refers to Chunk Refinement.

The DTL approach always results in learning more templates than the STL approach since for each input sentence pair, two difference templates and one similarity template can be derived. From Table 4, pure DTL systems (TExt-2 A and C) learned the fewest number of chunks from the corpus. The few chunks that are learned usually contain common words and are short. These chunks could match input sentences easily and thus result in learning a new template using the Derivation of Templates from Chunks (DTC) process. The implementation of chunk refinement in (TExt-2 C) resulted in more chunks being

learned, which in turn allowed more templates to be derived from input sentence pairs using DTC, thereby decreasing the number of unused sentences.

Chunk refinement, combined with STL (TExt-2 B and E), also resulted in an increase in the number of templates learned and decrease in the number of unused sentences. Because of refinement, long chunks are generalized into multiple smaller chunks which can easily be used by DTC to match against input sentences and learn new templates in the process.

Although the DTL algorithm was able to derive more templates compared to the STL algorithm, it made fewer calls to the Deriving Templates from Sentences (DTS) process. The effect is that the templates which were learned were not difference templates but rather templates which were derived from existing chunks.

Translating Sentences Using Similarity and Difference Templates

Corpus #5, which contains sentences derived from corpora #1-#4 used for training was also used to evaluate the translation quality of TExt-2. This corpus contains sentence patterns and words that match the training set. WER, SER, BLEU, and CWP (Correct Words Produced) are used to compare the translation quality of TExt-1 with the five variants

Table 4. Training Results to Extract Similarity and Difference Templates from Corpora #1 - #4

System	# of Templates Learned	# of Chunks Learned	# of Unused Sentences
TExt-1 - STL	73	210	29
TExt-2 (A) - DTL	95	92	34
TExt-2 (B) - STL with CR	78	229	24
TExt-2 (C) - DTL with CR	113	110	24
TExt-2 (D) - STL DTL	119	218	23
TExt-2 (E) - STL DTL with CR	133	253	18

of TExt-2. The results of translating documents from English to Filipino and from Filipino to English are shown in Tables 5 and 6, respectively.

The WER and SER scores of the two pure DTL systems (TExt-2 A and C) increased compared to the pure STL systems (TExt-1 and TExt-2 B). Although the DTL algorithm learned more templates, these additional resources did not contribute much to the improvement of the translation quality since the sentences in corpus #5 were designed to use the templates and chunks learned through STL.

For the same reason, none of the chunks generated through chunk refinement contained words that are found in the test sentences. Thus, CR did not result in any significant change in the scores.

Combining STL, DTL and CR (TExt-2 E) generated another set of scores. For the English to Filipino translation, the scores of the system with and without CR (TExt-2 D) are the same. For the Filipino to English translation, the chunks learned through CR affected the translation, thus increasing the error rate scores. This also shows that a chunk can be used in one language but not in the other. Consider an existing chunk [32], and the input sentences S26 and S27:

[32]: your kid « iyong anak

S26: Filipino Sentence:

Alagaan ang iyong anak.

S27: English Sentence:

Take care of your child.

Table 5. Test Results for English to Filipino Translation of Corpus #5

System	WER %	CWP %	SER %	BLEU
TExt-1 - STL	13.49	88.45	60.00	0.7470
TExt-2 (A) - DTL	43.25	65.18	86.67	0.4531
TExt-2 (B) - STL with CR	13.49	88.45	60.00	0.7470
TExt-2 (C) - DTL with CR	43.25	65.62	86.67	0.4553
TExt-2 (D) - STL DTL	15.17	87.32	73.33	0.7126
TExt-2 (E) - STL DTL with CR	15.17	87.32	73.33	0.7126

Table 6. Test Results for Filipino to English Translation of Corpus #5

System	WER %	CWP %	SER %	BLEU
TExt-1 - STL	18.12	86.00*	56.67	0.6990
TExt-2 (A) - DTL	55.49	60.79	83.33	0.3455
TExt-2 (B) - STL with CR	18.12	86	56.67	0.7027
TExt-2 (C) - DTL with CR	55.18	59.96	83.33	0.3440
TExt-2 (D) - STL DTL	21.85	82.66	63.33	0.6771
TExt-2 (E) - STL DTL with CR	27.37	77.59	66.67	0.6205

Chunk [32] can be used to translate the given Filipino sentence **S26** to English. However, it cannot be used to translate the given English sentence **S27** to Filipino since “**your child**” is not the same as “**your kid**”.

To test the translation quality of TExt-2 on an input document whose sentence patterns and words do not match the training set, corpus #6 containing 126 sentence pairs was derived from a document consisting of English-Filipino translations. The results of translating this corpus from English to Filipino and from Filipino to English are shown in Tables 7 and 8, respectively.

Combining STL, DTL and CR (TExt-2 E) produced the best English to Filipino translation quality with the lowest error rate scores. This could be due to the high number of chunks learned during

training (see Table 4). Since the sentences in corpus #6 are new to the system, that is, the patterns do not match the training set, chunks were used more than templates during translation. The availability of more chunks helped improve the translation quality for sentences whose patterns have not yet been learned by the system.

CONCLUSION

Learning translation patterns from examples provides a more flexible machine translation system that is less dependent on limited linguistic resources. TExt Translation is able to generate quality translation of English or Filipino documents, using templates and chunks that the system acquired during training from bilingual corpora. The

Table 7. Test Results for English to Filipino Translation of Corpus #6

System	WER %	CWP %	SER %	BLEU
TExt-1 - STL	89.96	29.70	100.00	0.0517
TExt-2 (A) - DTL	91.69	27.63	100.00	0.0299
TExt-2 (B) - STL with CR	90.29	30.17	100.00	0.0505
TExt-2 (C) - DTL with CR	92.02	28.00	100.00	0.0288
TExt-2 (D) - STL DTL	89.90	30.68	100.00	0.0523
TExt-2 (E) - STL DTL with CR	89.75	30.64	100.00	0.0528

Table 8. Test Results for Filipino to English Translation of Corpus #6

System	WER %	CWP %	SER %	BLEU
TExt-1 - STL	83.19	21.99	100.00	0.0322
TExt-2 (A) - DTL	85.46	21.73	100.00	0.0337
TExt-2 (B) - STL with CR	82.26	21.47	100.00	0.0333
TExt-2 (C) - DTL with CR	85.16	21.84	100.00	0.0339
TExt-2 (D) - STL DTL	80.78	21.51	100.00	0.0334
TExt-2 (E) - STL DTL with CR	83.11	21.60	100.00	0.0353

Strict Chunk Alignment with Splitting (SCAS) algorithm using Common Words Filtering (CWF) derived more templates and fewer chunks, which aid in producing better translation quality since sentence structures are captured and preserved. Moreover, the availability of more templates allow for refinement in subsequent training sessions.

The learning algorithm employed in this research also allows for chunks to be learned directly from the given examples, thus supplementing the available lexicon. These chunks are also utilized in deriving new templates from input sentences where no matching existing template can be found. Thus, chunks provide a solution to the constraints that templates impose on word ordering.

Tests showed that the implementation of a difference template learning algorithm combined with chunk refinement increased the number of templates that are learned from the corpus. However, this did not result in any significant improvement to the translation quality when the document to be translated contains sentence patterns and words that closely match the training set. In this case, the similarity template learning algorithm is sufficient. DTL, combined with STL and CR, showed much promise when the document to be translated is different from the corpus used in training the system.

Template refinement is a major component in the learning process of the TExt system. The addition of chunk refinement did not consistently show if this approach can provide better translation quality, although chunk refinement led to the generation of more templates from existing chunks during the training process and a corresponding reduction in the number of unused input sentences.

The approaches used in this research are highly dependent on the amount and quality of the corpus used for training and translation. Thus, additional corpora are needed to be able to study with greater depth the effects of DTL and CR. The corpora used were biased with learning and using the similarity templates. Very few sentences during translation were able to use difference templates since the sentences in the translation corpora are made to use similarity templates.

Some improvements can still be made to the system. Words are found to have different meanings depending on the context of the sentence that they occur in and consequently have different translations in the other language. In the current implementation, the first corresponding translation of a given word, regardless of context, is used even though another translation may be much more appropriate than the first.

The words that can be aligned by the current system are limited by the words in the lexicon and in the morphological lookup tables. Having a morphological analyzer would increase the likelihood of finding the alignments for the words in a sentence.

REFERENCES

- Ang, R., Bautisita, N., Cai, Y.R., & Tanlo, B. (2005). Translation with rule-learning. Undergraduate Thesis, College of Computer Studies, De La Salle University-Manila.
- Carl, M. (1999). Inducing translation templates for example-based machine translation. In *Proceedings of the Seventh Machine Translation Summit (MT-Summit VII)*, pp. 250-258.
- Cicekli, I. & Güvenir, H.A. (2003). Learning Translation Templates from Bilingual Translation Examples. In *Recent Advances in Example-Based Machine Translation*, Kluwer Publishers, Netherlands, pp. 255-286, 2003.
- In *Applied Intelligence*, 15(1), 57-76, 2001.
- Go, K., Morga, M., Nunez, V., Veto, F. & Ong, E. (2006). TExt Translation: Template Extraction for a Bidirectional English-Filipino Example-Based Machine Translation. Undergraduate Thesis, College of Computer Studies, De La Salle University, Manila.
- Go, K., Morga, M., Nunez, V. & Veto, F. (2007). Template-Based English-Filipino Machine Translation System. In *Proc. of the 4th National Natural Language Processing Symposium*, De La Salle University, Manila, pp. 43-47.

- Go, K., Morga, M., Nunez, V., Veto, F. & Ong, E. (2007a). Template Extraction for a Bidirectional English-Filipino Machine Translation System. In *Proc. of the 7th Philippine Computing Science Congress*, Manila.
- Hovy, E., Ide, N., Frederking, R., Mariani, J. & Zampolli, A. (2001). Multilingual Information Management: Current Levels and Future Abilities. In *Linguistica Computazionale*, Volume XIV-XV, Italy. Available online: <http://www.cs.cmu.edu/~ref/mlim/index.html>
- Kaji, H., Kida, Y., & Morimoto, Y. (1992). Learning Translation Templates from Bilingual Text. In *Proc. 14th International Conference on Computational Linguistics (COLING-92)*, Volume 2, France, pp. 672-678.
- Kinoshita, S., Kumano, A., & Hirakawa, H. (1994). Improvement in Customizability Using Translation Templates. In *Proc. 15th International Conference on Computational Linguistics (COLING-94)*, Volume 1, Japan.
- Nunez, V. & Ong, E. (2007). Combining Similarity and Difference Templates for a Bidirectional Example-Based Machine Translation. Graduate Thesis, College of Computer Studies, De La Salle University, Manila.