International Journal of Engineering Inventions
e-ISSN: 2278-7461, p-ISSN: 2319-6491
Volume 3, Issue 2 (September 2013) PP: 71-81

How to Translate from English to Khmer using Moses

Suraiya Jabin[1], Suos Samak[1], Kim Sokphyrum[2]
[1]*Department of Computer Science, Jamia Millia Islamia, Central University, New Delhi, India*
[2]*Department of Computer Science, University of Delhi, New Delhi, India*

ABSTRACT: This paper presents systematic and step by step methodology for designing an online English to Khmer machine translation system using Moses DoMY CE, an open source Statistical Machine Translation (SMT) tool. The usefulness of this paper is that the same methodology can be adopted to produce an online SMT between any language pair. Moses is an implementation of the statistical (or data-driven) approach to machine translation. This is the dominant approach in the field at the moment, and is employed by the online translation systems like Google and Microsoft. We have attempted to minutely explain each step including formation of input files and output files in each step taken by Moses.

Keywords: Corpus, Khmer, Moses, Phrase-based model, Statistical Machine Translation

I. INTRODUCTION

Machine Translation (MT) can be defined as the use of computers to automate some or all of the process of translating from one language to another. MT is an area of applied research that draws ideas and techniques from linguistics, computer science, Artificial Intelligence (AI), translation theory, and statistics. Work began in this field as early as in the late 1940s, and various approaches; some ad hoc, others based on elaborate theories; have been tried over the past five decades. The statistical approach to MT, which was first suggested by Warren Weaver in 1949 [3], has found practical relevance only in the last decade or so. This approach has been made feasible by the vast advances in computer technology, in terms of speed and storage capacity, and the availability of large quantities of text data. Statistical machine translation utilizes statistical translation models whose parameters stem from the analysis of monolingual and bilingual corpora. Building statistical translation models is a quick process, but the technology relies heavily on existing multilingual corpora. A minimum of 2 million words for a specific domain and even more for general language are required. Theoretically it is possible to reach the quality threshold but most companies do not have such large amounts of existing multilingual corpora to build the necessary translation models. Proposed system can be treated as a prototype as we trained it on some 5000 training examples. Additionally, statistical machine translation is CPU intensive and requires an extensive hardware configuration to run translation models for average performance levels.

Khmer, is an official language of Cambodia, influenced by Sanskrit and Pali languages combining features of Hinduism and Buddhism. Khmer is primarily an analytic, isolating language. There are no inflections, conjugations or case endings. Instead, particles and auxiliary words are used to indicate grammatical relationships. General word order is subject–verb–object. Many words conform to the typical Mon-Khmer pattern, having a "main" syllable preceded by a minor syllable. The Khmer language is written with an abugida known in Khmer as អក្ខរក្រម. Khmer differs from neighboring languages such as Thai, Burmese, Lao and Vietnamese in that it is not a tonal language [1].

Most statistical machine translation (SMT) research has focused on a few high-resource languages (European, Chinese, Japanese, Arabic). The major of the educational text in Cambodia is available in English, so there is a rising demand of English to Khmer translation. As we aimed to provide everyone in Cambodia with access to all of the world's information, mainly information written in English, we undertook this exciting project of English to Khmer translation. We have used a parallel corpus which is very small as compared to the parallel corpora used to train a SMT system. But we are enhancing our parallel corpora so that proposed system's performance will further improve. At present state we can assume it as a prototype of English to Khmer SMT system. A step by step method is demonstrated to build a **S**tatistical **M**achine **T**ranslation (SMT) system from English to Khmer.

II. MOSES DOMY CE

Do Moses Yourself™ Community Edition (DoMY™ CE) transforms Moses into a simple do-it-yourself system, and we *immediately* enjoy the benefits of translation software that works. The Moses Decoder statistical machine translation (SMT) toolkit is the foundation of Do Moses Yourself (DoMY™). The figure 1 shows architecture of Moses Decoder:

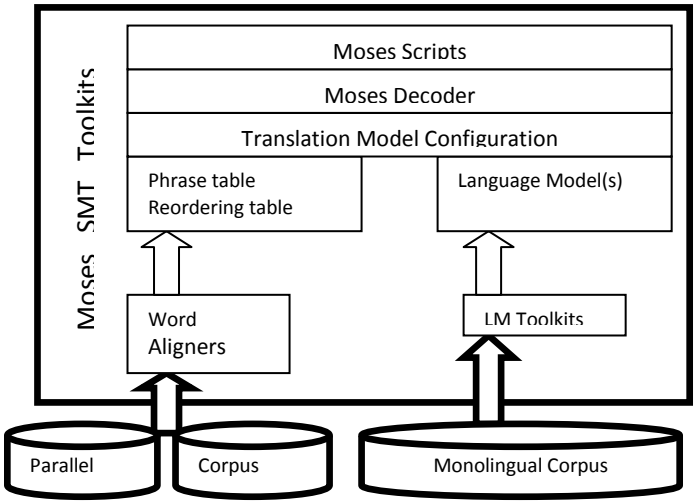


Figure 1: Architecture of Moses Decoder [4]

Moses is an implementation of the statistical (or data-driven) approach to machine translation. This is the dominant approach in the field at the moment, and is employed by the online translation systems deployed by Google and Microsoft. In statistical machine translation (SMT), translation systems are trained on large quantities of parallel data (from which the systems learn how to translate small segments), as well as even larger quantities of monolingual data (from which the systems learn what the target language should look like).
Parallel data is a collection of sentences in two different languages, which is sentence-aligned, in that each sentence in one language is matched with its corresponding translated sentence in the other language [4].
The training process in Moses takes in the parallel data and uses co-occurrences of words and segments (known as phrases) to infer translation correspondences between the two languages of interest. In phrase-based machine translation, these correspondences are simply between continuous sequences of words.
These are the main features of Moses:

- Moses offers two types of translation models: phrase-based and tree-based. We have used phrase based model for the proposed system.
- Moses features factored translation models, which enable the integration linguistic and other information at the word level.
- Moses allows the decoding of confusion networks and word lattices, enabling easy integration with ambiguous upstream tools, such as automatic speech recognizers or morphological analyzers.
- The Experiment Management System further simplifies and makes it much easier using Moses.

These are the following main steps taken by Moses SMT toolkit in order to do translation between any language pair [4]:

- Moses training scripts (blue) use word aligners to generate phrase and reordering tables.
- Language model toolkits (green) generate word order probabilities in the target language.
- Translation matches word sequences in source sentences to target word sequences in the tables. Then, Moses uses the word order probabilities to select the best possible match.
- Tuning repeatedly translates source sentences, compares the results to reference sentences, and finds the optimal configuration for the best translation output.
- Evaluation translates source sentences using the optimal configuration and generates an evaluation report that includes BLEU and NIST scores for final review.

III. STEP BY STEP METHOD FOR TRANSLATING ENGLISH TO KHMER

Moses is a statistical machine translation system that allows you to automatically train translation models for any language pair. All you need is a collection of translated texts (parallel corpus). An efficient search algorithm finds quickly the highest probability translation among the exponential number of choices. The first step is to install Moses on a very high speed machine with UNIX platform.

3.1 DoMY CE Installation

In order to install DoMY CE, follow the following steps:

(a) Add "PPA for Do Moses Yourself (DoMY CE)" by type the following command in terminal:
~$: sudo add-apt-repository ppa:support-precisiontranslationtools/domosesyourself
(b) Update your system
~$: sudo apt-get update
(c) Install Do Moses Yourself Community Edition (DoMY CE) package
~$: sudo apt-get install domy-ce

The packages such as automake, autoconf, build-essential: libc6-dev, libc-dev, g++, make, dpkg-dev, bzip2, csh, gawk, gzip, libboost-all-dev, libtool, libxmlrpc-c3-dev, p7zip, subversion, tcl-dev, wget, xz-utils, zip, zlib1g-dev, zlib1g are automatically installed with it.

This is the removal command for DoMY CE:
~$: sudo apt-get remove domy-ce

Locations where DoMY CE files are installed:

- **DoMY CE/CorpusFiltergraph system folder:** /usr/local/lib/corpusfg
- **Master graphs folder:** /usr/local/lib/corpusfg/graphs
- **Master plug-ins folder:** /usr/local/lib/corpusfg/plugins
- **Configuration files with host default:** /usr/local/etc/corpusfg
- **Configuration files for translation and recaser models:** /usr/local/etc/domy
- **Package installation logs:** /var/log/<package>
- **Status logs:** /var/log/corpusfg/logs
- **Root folder for corpora data, language models, tables, translation models and temporary support files, manual doc, glossary and other documentation files:** /opt/domy
- **Configuration files with user default:** /home/<user>/domy
- **User copy of graphs:** /home/<user>/domy/graphs
- **User copy of plug-ins:** /home/<user>/domy/plugins

3.2 Data Preparation

To train a translation system we need parallel data (text translated into two different languages) which is aligned at the sentence level. We aligned English-Khmer corpus, sentence by sentence per line in separate files. For instance, in English corpora, the first line is ***"lesson 1",*** and then Khmer corpora must contain its corresponding sentence ***"មេរៀន ទី ១ [Merean Ty Muoy]"***. Full stop ***"."*** is the delimiter of English sentences and ***"។"*** for Khmer sentences.

For the proposed system, we used five thousand seven hundred and thirty four (5734) sentences for English-Khmer Corpus. English and Khmer corpus must be saved using the same name but stored in two different folders namely English and Khmer respectively. Also both English and Khmer corpora text files must be saved in **"utf-8"** encoding standard. These corpus are stored in directories as the following:

For **English** : ***/opt/domy/CORPORA/sa/Eng-Kh/tm/en/en/en.txt*** and
For **Khmer** : ***/opt/domy/CORPORA/sa/Eng-Kh/tm/en/kh/en.txt***

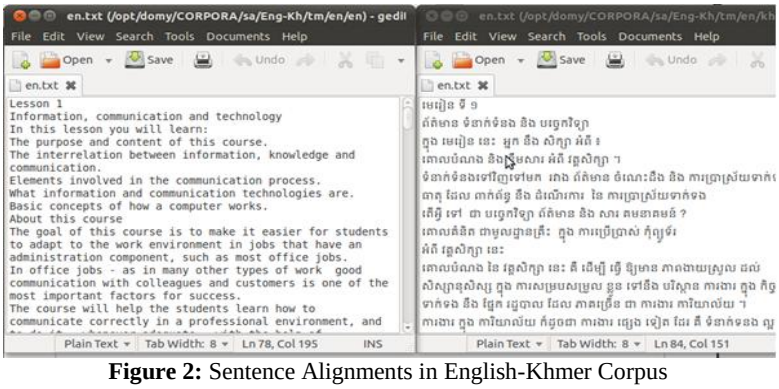


Figure 2: Sentence Alignments in English-Khmer Corpus

3.3 Data Cleaning

Data cleaning is the process of removing long sentences and empty sentences as they can cause problems with the training pipeline, and obviously misaligned sentences are removed. For cleaning or preparing the sentences with the following two conditions, use the command: $ ***domy clean-tm***

- Sentence pairs having count of words count within range are saved to the **"CORPORA/ready/"** to be used as SMT training data
- Sentence pairs having count of words outside the range are saved to the **"CORPORA/ready-workbench/"** to be used for review by qualified editors

These two conditions depend on the range defined by values of **"mintoken"** and **"maxtoken"** in ***config.ini*** in **"/mt/domy/graphs/clean-tm/"**

Note: in the config.ini of clean-tm graph
mintoken=2
maxtoken=65

It will produce output file consisting of **sentences where count of words is in range 2 to 65 words.** This output file will be stored at location ***/opt/domy/CORPORA/ready/Eng-kh/tm/en*** . The sentences with count **of words outside this range** will be stored at location: ***/opt/domy/CORPORA/ready-workbench/Eng-Kh/tm/en.*** These files are shown in figure 5.

config.ini of clean-tm graph
/home/mt/domy/graphs/clean-tm

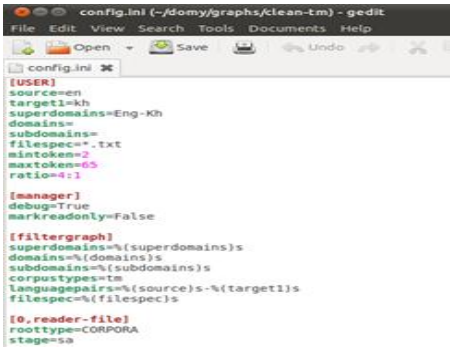


Figure 3: A portion of config.ini file of clean-tm graph.

Input files are the parallel corpus with English and its corresponding Khmer file. Each file is stored in separate folder as shown below. But both English and Khmer files must have the same name.

Location of Input files: ***/opt/domy/CORPORA/sa/Eng-kh/tm/en/en***

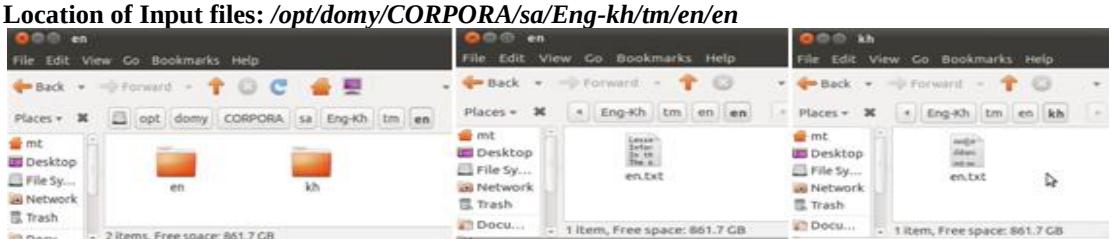


Figure 4: Input files to be cleaned

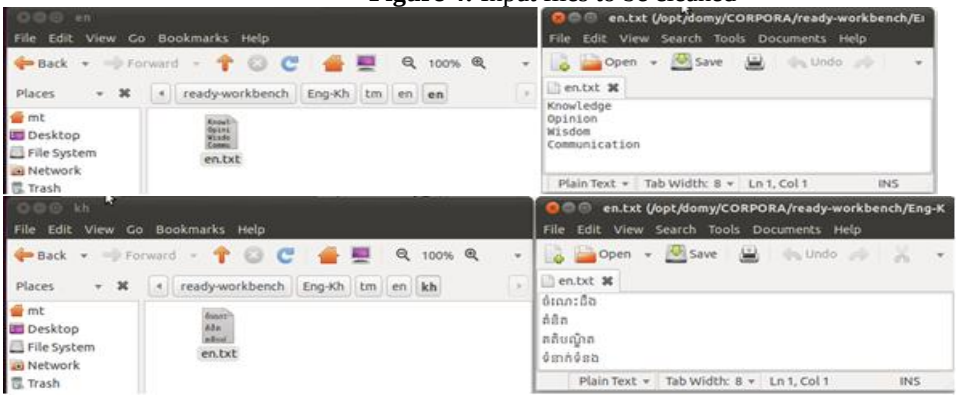


Figure5: Output files after cleaning

3.4 Build-tm

This command is used for tokenizing i.e. separating words from special characters by inserting spaces between words and punctuation. It also converts the initial word in each sentence into lower-case to avoid data sparsity. These operations work according to the ***filter plug-in option*** which we set in **config.ini** in the ***[%(source)s]*** and ***[%(target1)s]*** *section. And it is also used to merge input data from multiple files into a tm build set.*

Note: *DoMY CE application automatically replaces "zero-width-space" with "white space" in Khmer corpora.*
command: ***$ domy build-tm***

Examples:

Input:

Information, communication and technology
In this lesson you will learn:
The purpose and content of this course.
The interrelation between information, knowledge and communication.

Output Result:

information , communication and technology
in this lesson you will learn :
the purpose and content of this course .
the interrelation between information , knowledge and communication .

config.ini of build-tm graph

/home/mt/domy/graphs/build-tm

Input folder of English corpora

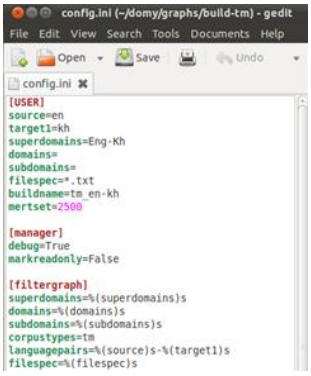
/opt/domy/CORPORA/ready/Eng-Kh/tm/en/en

and Input folder of Khmer corpora

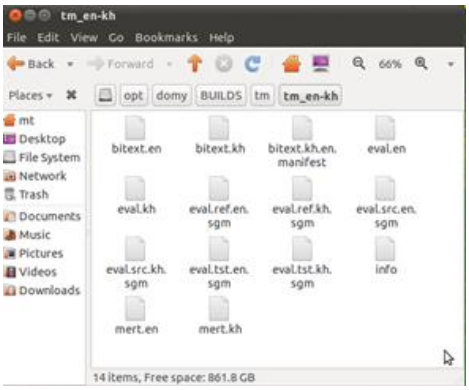
/opt/domy/CORPORA/ready/Eng-Kh/tm/en/kh

Output folder after running build-tm command

/opt/domy/BUILDS/tm/tm_en-kh/



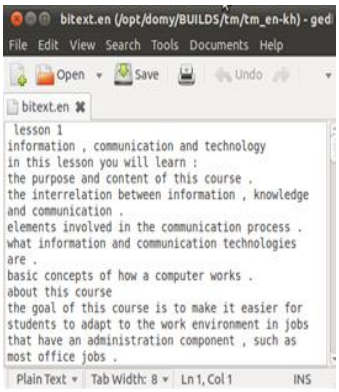
config.ini file of build-tm graph



Location output files of build-tm graph

Figure 6: build-tm

bitext.en: is the output file of English language obtained after running command build-tm
bitext.kh: is the output file of Khmer language obtained after running command build-tm
eval.en: contains one sentence from English language and
eval.kh: contains one corresponding Khmer sentence. These files are used for evaluation purpose.
eval.ref.en.sgm and **eval.ref.kh.sgm:** store the evaluation reference set of English-Khmer sentence in xml format.



bitext.en output files of build-tm graph

bitext.kh output files of build-tm graph

eval.en and eval.kh output files of build-tm graph

Figure 7: build-tm output

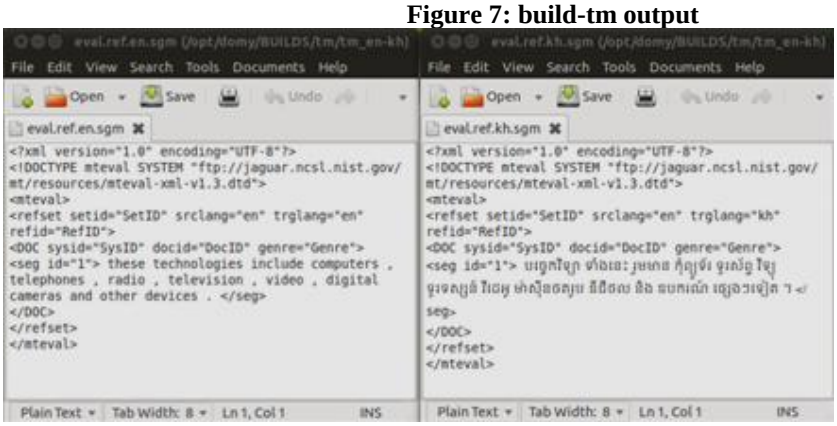


Figure 8: eval.ref.en.sgm and eval.ref.kh.sgm output files of build-tm graph

eval.src.en.sgm and ***eval.src.kh.sgm*** store the evaluation **source set** of English-Khmer sentence in xml format.

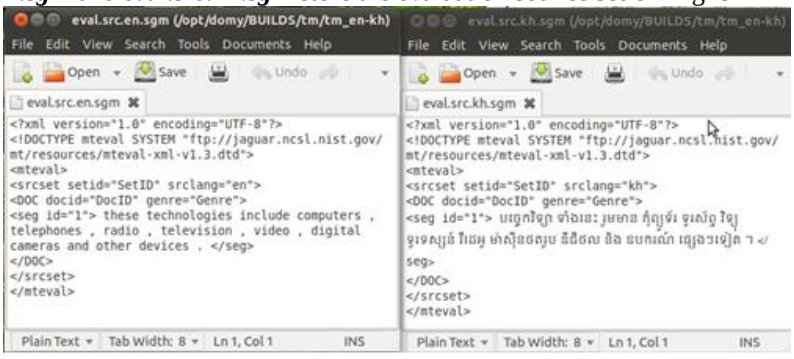


Figure 9: eval.src.en.sgm and eval.src.kh.sgm output files of build-tm graph

eval.tst.en.sgm and ***eval.tst.kh.sgm*** store the evaluation **target set** of English-Khmer sentence in xml format.

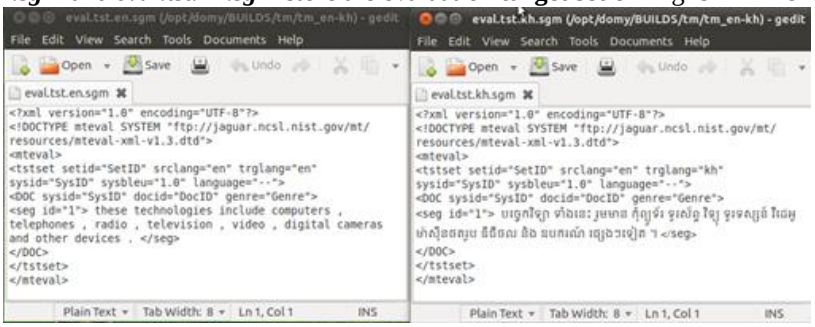
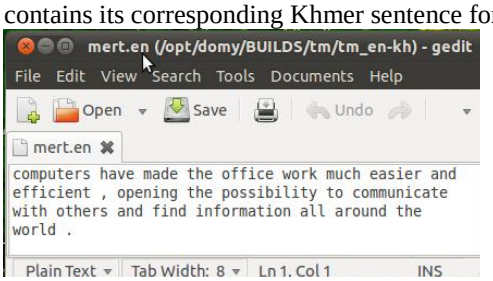


Figure 10: eval.tst.en.sgm and eval.tst.kh.sgm output files of build-tm graph

mert.en: contains another English sentence for evaluation purpose.
mert.kh: contains its corresponding Khmer sentence for evaluation purpose.



mert.en output files of build-tm graph

mert.kh output files of build-tm graph

Figure 11: mert

3.5 Build-lm

This command is used to consolidate multiple input data files into single language model build set. The output files consists of "**<s>"** (at the beginning of sentence) and "**</s>"** (at the end of one sentence) in order to identify individual sentence.
command: ***$ domy build-lm***

Example:

a. For English language (monotext.en)

Input sentence:

Lesson 1
Information, communication and technology
In this lesson you will learn:

Output sentence:

<s> lesson 1 </s>
<s> information , communication and technology </s>
<s> in this lesson you will learn : </s>

b. For Khmer language (monotext.kh)

Input sentence:

មេរៀន ទី ១ [***Merean Ty Muoy***]

ព័ត៌មាន ទំនាក់ទំនង និង បច្ចេកវិទ្យា [***Pordmean Tamnak Tamnong Ning Bakchekvitya***]

ក្នុង មេរៀន នេះ អ្នក នឹង សិក្សា អំពី ៖ [***Knong Merean Nis Neak Neng Seksa Amby***]

Output sentence:

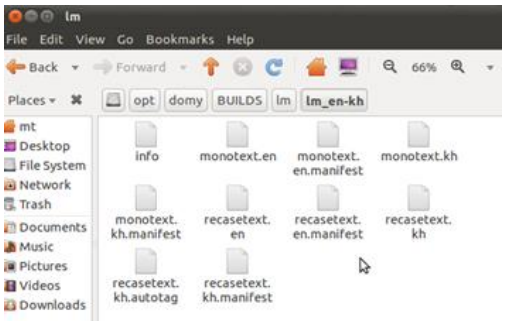
<s> ***មេរៀន ទី ១*** </s>

<s> ***ព័ត៌មាន ទំនាក់ទំនង និង បច្ចេកវិទ្យា*** </s>

<s> ***ក្នុង មេរៀន នេះ អ្នក នឹង សិក្សា អំពី ៖*** </s>

config.ini of build-lm graph
/home/mt/domy/graphs/build-lm

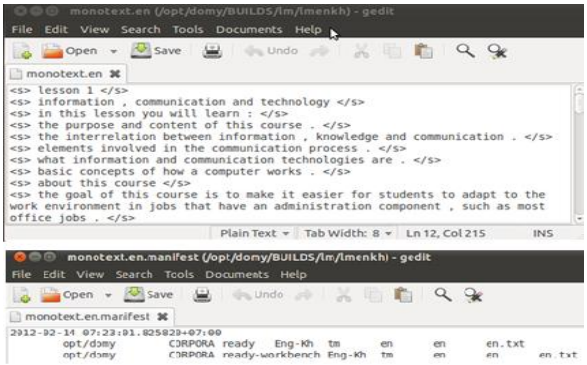
config.ini of build-lm graph



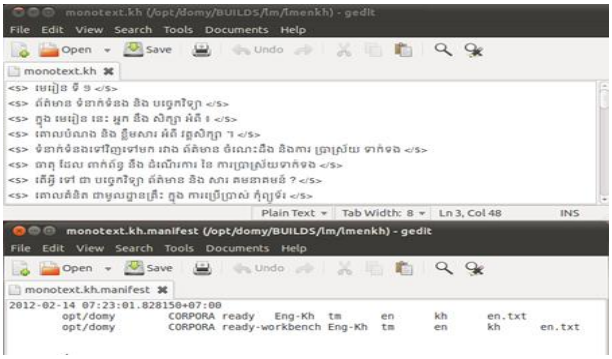
The output files of build-lm graph

Figure 12: build-lm Graph

Input folder which contains several English-Khmer parallel corpus files:
/opt/domy/CORPORA/ready/Eng-Kh/tm/en/en/.txt*
/opt/domy/CORPORA/ready-workbench/Eng-Kh/tm/en/en/.txt*
Output folder which combines several English files into single English file and the same for Khmer files:
/opt/domy/BUILDS/lm/lm_en-kh/
The following 8 files have been produced in this process:
"monotext.en" and "monotext.en.manifest"
"monotext.kh" and "monotext.kh.manifest"
"recasetext.en" and "recasetext.en.manifest"
"recasetext.kh" and "recasetext.kh.manifest"

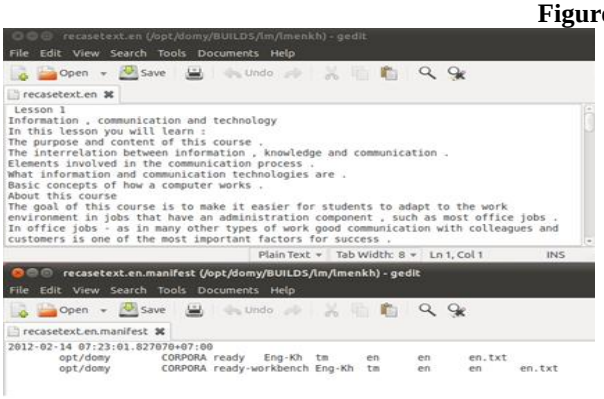


monotext.en" and "monotext.en.manifest" output files of build-lm graph

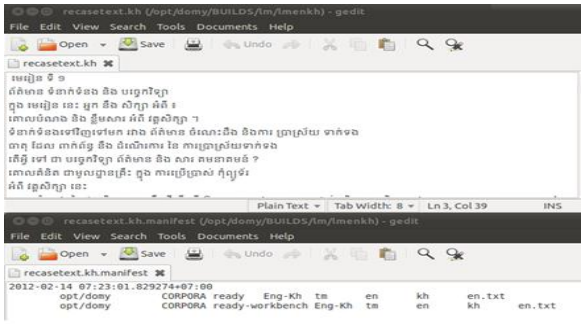


"monotext.kh" and "monotext.kh.manifest" output files of build-lm graph

Figure 13: build-lm



"recasetext.en" and "recasetext.en.manifest" output files of build-lm graph



"recasetext.kh" and "recasetext.kh.manifest" output files of build-lm graph

Figure 14: build-lm output

3.6 Training the Data

Training of the data means "*transform the prepared training data into the statistical models and configuration files, which are the components that constitute an SMT translation model."*

There are *5 types of train command:*

- ***train-lm:*** *train and binarize (irstlm, srilm, randlm) language model.*
- ***train-tables:*** produce phrase and reordering tables.
- ***train-mert:*** tune a translation model to find optimized configuration
- ***train-eval:*** evaluate tuned translation model configuration
- ***train-recaser:*** create a recaser model

First some modifications were made in the config.ini of train commands,
+ config.ini of "train (all)" and "train-eval"

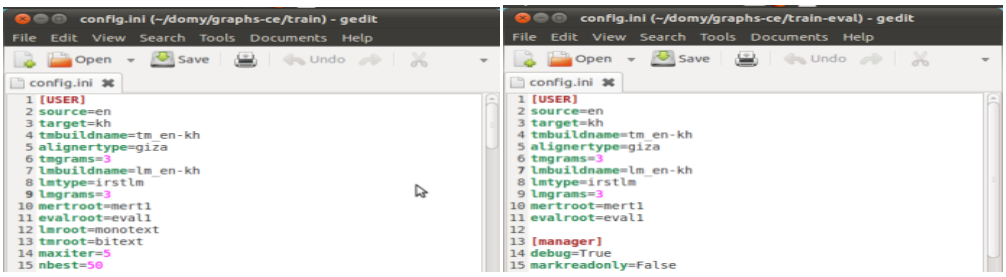


Figure 15: *config.ini of "train (all)" and "train-eval"*

+ config.ini of "train-lm" and "train-mert"

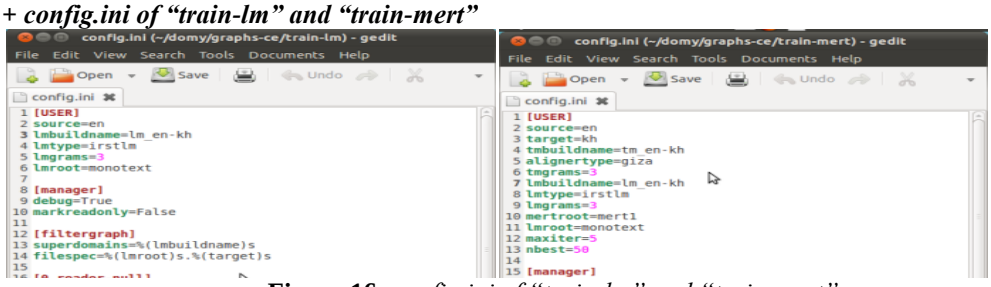


Figure 16: *config.ini of "train-lm" and "train-mert".*

+ config.ini of "train-recaser" and "train-tm"

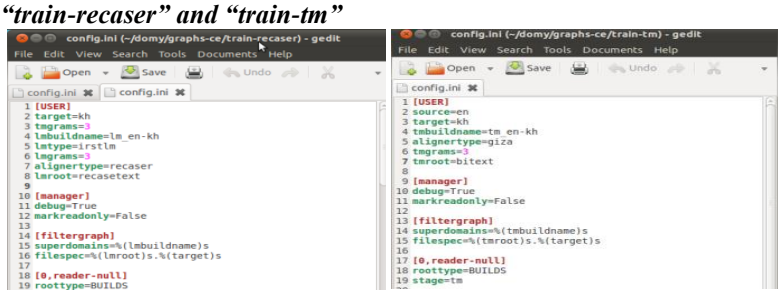


Figure 17: config.ini of "train-recaser" and "train-tm".

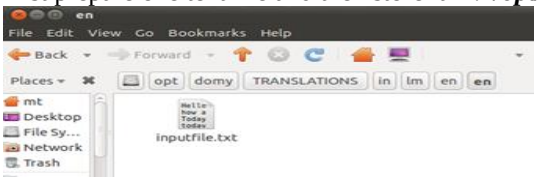
Then run command: ***domy train***
If we want to train on previous versions of DoMY CE we have to run command **"train-"** but for the latest version, the command is **"domy train".**

3.7 Translation

We used phrase-based decoder in Moses. Phrase-based translation models are acquired from a word-aligned parallel corpus by extracting all phrase-pairs that are consistent with the word alignment. Given the set of extracted phrase pairs with counts, various scoring functions are estimated, such as conditional phrase translation probabilities based on relative frequency estimation or lexical translation probabilities based on the words in the phrases. In Moses, the models for the translation steps are acquired in the same manner from a word-aligned parallel corpus. For the specified factors in the input and output, phrase mappings are extracted. The set of phrase mappings (now over factored representations) is scored based on relative counts and word-based translation probabilities.

a. Input English text file

First prepare one text file and then store it in: ***/opt/domy/TRANSLATIONS/in/lm/en/en/inputfile.txt***



Location of Input English text file for translation process

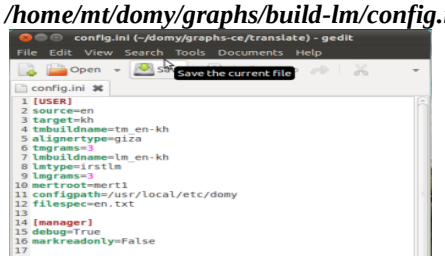
Input English text file for translation process

Figure 18: Input English Text

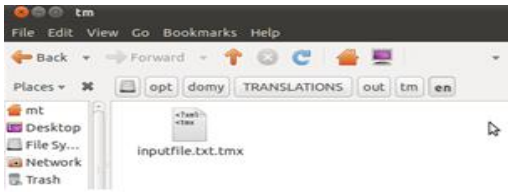
b. Translate

The command: **$ domy translate** is used to produce output of translate file from English to Khmer Language. Before running this command we have to change some values in **"config.ini"**

/home/mt/domy/graphs/build-lm/config.ini



config.ini file of translate graph



Location of output file after the translating

Figure 19: Output File

c. Output file (Khmer translated output file)

*This output file is in **"TMX" format**. It stores in **/opt/domy/TRANSLATIONS/out/tm/en/***

Figure 20: *The content of TMX file.*

3.8 Web technologies used

As we designed our system as a web-based system, we used Apache2 web server for delivering our system. Apache is the most commonly used Web server on Linux Systems. Web Servers are used to serve Web Pages requested by client computers. Clients typically request and view web pages using Web browser applications such as Firefox, Opera, and Internet Explorer (IE).

Method to install Apache2 web server in Ubuntu 10.10

Make the installed program up-to-date by running the following commands:

- **sudo apt-get update**
- **sudo apt-get upgrade**

Install Apache2 web server

Type command: **$ sudo apt-get install apache2 apache2-doc apache-utils**

Installing support for scripting

The following commands are optional and should be run if you want to have support within Apache for server-side scripting in PHP, Ruby, Python, or Perl.

To install Ruby support

Command: **$ sudo apt-get install libapache2-mod-ruby**

To install Perl support

Command: **$ sudo apt-get install libapache2-mod-perl2**

To install Python support

Command: **$ sudo apt-get install libapache2-mod-python**

To install Mysql in Python support

Command: **$ sudo apt-get install python-mysqldb**

To install PHP5 support
Command: **$ sudo apt-get install libapache2-mod-php5 php5 php-pear php5-xcache**
To install MySQL in PHP support
Command: **$ sudo apt-get install php5-mysql**
Restart Apache2 Services
Command: **$ sudo /etc/init.d/apache2 restart**

3.9 Output Screen

This web-based application is developed using python web CGI technology and HTML. It runs on Appache2 web server. Given Input text (English), it displays system translated Khmer text. The main problem we observed was preparation of parallel corpus from the scratch. We have obtained good performance results compared with Google Translate for in-domain sentences.

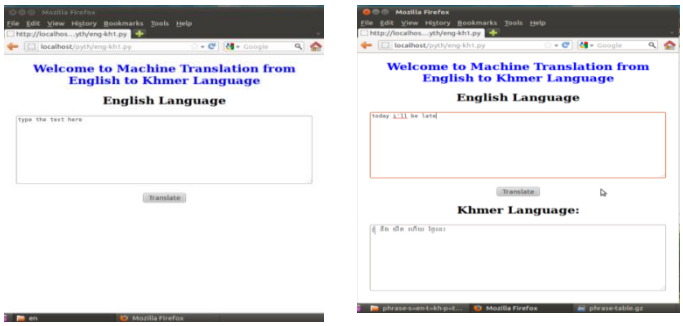


Figure 21: Homepage of Proposed English to Khmer SMT system

IV. CONCLUSION

A step by step methodology is presented for preparing English to Khmer SMT system using Moses. The Moses toolkit consists of all the components needed to preprocess data, train the language models and the translation models. It contains tools for tuning these models and automatically evaluating the resulting translations. The parallel corpus used in our experiments is very small (only 5000 sentences) but as far as we know there is no ready-made corpus available for English-Khmer. Adding more sentences in the used parallel corpus, will certainly enhance the performance of the proposed system towards more generalized kind of translation. Further attempts will be taken to incorporate example-based machine translation method which retrieves similar examples (pairs of source phrases, sentences, or texts and their translations) from a database of examples, adapting the examples to translate a new input.

V. ACKNOWLEDGEMENTS

Prof. Niladri Chatterjee, Prof. Vasudha Bhatnagar and Mr. Javier Sola are names that deserve special mention; as they provided informative guidance and motivation for the proposed work.

REFERENCES

[1.] Source: http://en.wikipedia.org/wiki/Khmer_language
[2.] Moses: Statistical Machine Translation System, User Manual and Code Guide by Philipp Koehn, www.statmt.org/moses/manual/manual.pdf
[3.] Warren Weaver, Translation, Machine Translation of Languages: Fourteen Essays, William Locke and Donald Booth (eds), pages 15–23, 1955.
[4.] Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In ACL Companion Volume. Proc. of the Demo and Poster Sessions, pages 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics. www.statmt.org/moses