STATISTICAL MACHINE TRANSLATION USING MOSES DECODER

Devashree S. Shinde,
Department of Computer,
MMCOE, Pune-411052.
devashree28@gmail.com

Shweta S. Porwal,
Department of Computer,
MMCOE, Pune-411052.
shweta.s.porwal@gmail.com

Rachana R. Athalekar,
Department of Computer,
MMCOE, Pune-411052.
rachana90@gmail.com

Yugesha R. Sapte
Department of Computer,
MMCOE, Pune-411052.
yugesha.sapte4@gmail.com

***Abstract*— The dissimilarities between various languages around the globe and in particular the inherent ambiguity of languages make human translation very difficult. Machine Translation simplifies the process of translation. One of the most promising machine translation strategies would be Statistical Machine Translation (SMT) approach. Being relevant even to structurally dissimilar language pairs, SMT becomes suitable for large text translations. A basic translation system which uses an open source toolkit, Moses, has been described in this paper with the preparation of parallel bilingual corpus. This corpus is incorporated as a core element of Moses. Ongoing research in the field of machine translation highlights Moses as a tool which will open up new avenues for development in this domain.**

***Index Terms* — Parallel bilingual corpus, Moses, Statistical Machine Translation.**

I. INTRODUCTION

Language is the main form of human communication. Translation is essential for co-operation among communities that speak different languages. Machine Translation refers to the use of computers to automate the task of translation between human languages. In order to perform such a task, the computer must know the two languages which form the specified language pair. One of the crude ways to incorporate this knowledge into a machine is to use language experts to program the necessary information into it. This creates a barrier as many of the components are required to be duplicated. A more refined way is to let the machine learn some of these concepts automatically by examining large amounts of parallel bilingual corpus.

There are various approaches to achieve machine translation in this form. Some of them being:

i) Rule-based machine translation
ii) Example-based machine translation
iii) Statistical machine translation
iv) Hybrid machine translation

i) **Rule-based machine translation** (**RBMT):**
It is a methodology based on linguistic information about source and target languages retrieved from bilingual dictionaries and grammars covering the main semantic aspects of each language.

ii) **Example-based machine translation (EBMT):**
It is a methodology which is characterized by its use of a bilingual corpus with parallel texts as its main knowledge base, at run-time.

iii) **Statistical machine translation (SMT):**
In this methodology, translations are generated on the basis of statistical models whose parameters are derived from the analysis of parallel bilingual corpus. In this, there is no consultation of the corpus at run-time, as all data is derived in advance of the translation process.

iv) **Hybrid machine translation (HMT):**
It is a methodology, which exploits the strengths of statistical and rule-based translation.

This paper focuses on the statistical approach for machine translation. Moses is an open-source toolkit which provides a platform for implementing statistical machine translation.

Section II gives an overview of statistical machine translation. Section III highlights the acquisition and processing of parallel bilingual corpus. This corpus forms the basis on which Moses builds its models. Section IV describes the detailed architecture of the Moses toolkit. It also explains the basic mathematical principle on which Moses works. Section V highlights the nuances of the entire experimental setup which brings together all the components of the proposed statistical machine translation system.

II. STATISTICAL MACHINE TRANSLATION

A. Overview

Statistical Machine Translation (SMT) models are based on the assumption that every sentence in the target language is a translation of a sentence in the source language with some probability. The best translation is the sentence that has the highest probability. [1][3]

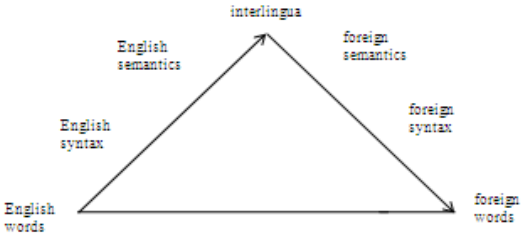

Fig. 1. Translation process

Few of the reasons why SMT is preferred over other approaches to machine translation:

i) Removes Fundamental uncertainties (e.g. ambiguity)
ii) With the large no of corpus collection, sometimes whole sentences can be covered
iii) Few linguistic assumptions
iv) Applicability to any language pair, with large corpora
v) Shorter development time [7]

B. Basic mathematics

In statistical machine translation, we assume that every target language string, T, is a possible translation of S. Every pair of string (S, T) is assigned a number Pr (T|S), which is the probability that the translator, when presented with S, will produce T as its translation. The possibility of error is minimized by choosing that sentence S that most probably gives T. Thus, we must choose S so as to maximize Pr(S|T). Therefore, using Bayes' theorem, we get

$$\Pr(S|T) = \frac{\Pr(S).\Pr(T|S)}{\Pr(T)}$$

The denominator on the right of the above equation does not depend on S. So it is sufficient to choose the S that maximizes the product Pr(S).Pr(T|S). The first factor in this product is the language model probability of S and the second factor is the translation probability of T, given S. [4]

Baye's Theorem is used in the Language Model (LM) of Moses decoder.

III. COLLECTION OF PARALLEL BILINGUAL CORPUS

Development in natural language processing is driven by the availability of data. Statistical machine translation thrives on large quantities of parallel text, which is text paired with its translation to another language. This parallel text can often be available with multinational institutions.

This parallel corpus can be acquired for use in a statistical machine translation system by the means of the following five steps:

i) obtain the raw data by using a web crawler
ii) extract and map chunks of data into parallel text
iii) split the text into sentences
iv) prepare the corpus for SMT systems by normalization and tokenization
v) map sentences from source language to target language

i) **Obtain the raw data by using a web crawler**:
Web crawlers are used to gather the data from webpages by specifying the desired URLs. As the crawler visits these URLs, it identifies all the hyperlinks in the page and adds them to the list of URLs to visit thereby, obtaining maximum amount of data possible from the specified URL.

ii) **Extract and map chunks of data into parallel text:**
The raw data may cover several topics. The purpose of this step is to identify the texts belonging to each topic and matching these between languages. This is done without tokenization and sentence splitting.

iii) **Split the text into sentences:**
The data obtained in the above steps is in the form of paragraphs. These paragraphs need to be broken down into sentences.

iv) **Prepare the corpus for SMT systems by normalization and tokenization:**
Tokenization includes merging of words (e.g. can't has to be converted to cannot), removal of possessive markers (e.g. man's becomes man 's), removal of punctuations. The sentences need to be converted into lowercase which is known as normalization.

v) **Map sentences from source language to target language:**
After normalization and tokenization, the sentences are aligned into source and target language pairs respectively. [5]

The result of the above steps will appear as follows:

Source Language	Target language
can we meet today	können wir heute begegnen
no we cannot meet	nein wir nicht erfüllen können
we will meet some other day	treffen wir uns an einem anderen tag

This parallel bilingual corpus acts as an input to Moses for further processing.

The quality of the translation depends on the amount of the content available online for the required language pair. More frequently used languages have enormous amount of content available. Hence translation quality of these languages is very high. On the contrary, languages that are not frequently used have a very limited content available which can lead to imprecise translations. Therefore, larger the corpus, larger is the accuracy of the translated sentence.

IV. MOSES

The Moses architecture follows a very modular design. The following diagram illustrates the various blocks that form a part of the toolkit. [6]

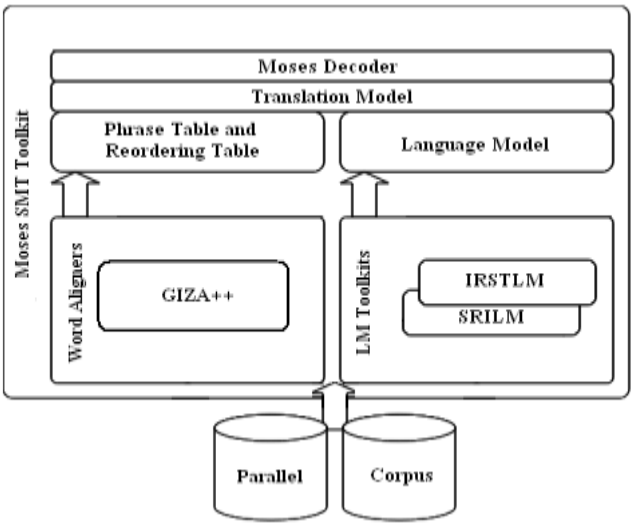

Fig.2. Block diagram of Moses

Language Model:
The language model in Moses assigns a probability to a sequence of words in the source language. This is done by probability distribution. Relating this to Bayes' theorem, language model performs the task of calculating Pr(S). In the collected bilingual parallel corpus, the included phrases and sentences can be arbitrarily long, making it difficult to estimate their probabilities. Hence, these probabilities need to be approximated during the language model training. N-gram models provide a way of doing this. This model in Moses makes use of IRSTLM and SRILM toolkits.

Translation Model:
The translation model assigns a conditional probability to the words in target language with respect to the source language. Relating this to Bayes' theorem, translation model performs the task of calculating Pr(T|S). It consists of 2 passes. In the 1st pass, it calculates the length of the string. In the 2nd pass, it searches the appropriate target word that needs to be replaced for the source word.

Phrase table and Reordering table:
Moses performs translations in a phrase-based manner. During decoding, the source language input sentence S is segmented into a sequence of 'p' phrases. A uniform probability distribution is achieved over all possible segmentations. The phrases in the target language may need to be reordered. Re-ordering of the words is required to ensure the syntactical structure as well as the grammatical construct of the sentence. In Moses, reordering is carried out with the help of GIZA++. The use of phrase table for translation is beneficial when one word in the source language translates into multiple words in the target language.

Example:
Processing of the following sentence through Moses.
"I went to the cinema house on Sunday."

Step I : Language Model
It calculates the probability of words using n gram models.
N gram is a collection of n words.

Bigram- Collection of 2 words
P ("I went")
P ("went to")
P ("to the")
and so on.

Trigram – Collection of 3 words
P ("I went to")
P ("went to the")
P ("to the cinema")
and so on.

Step II: Translational Model
Pass1: Calculates length of input sentence i.e. no of words present in it.
Length of input sentence = 8

Pass2 Find appropriate target word that needs to be replaced for the source word by calculating probability of same words from the corpus i.e. P (G/E).

Consider the word "house". Following are the words available in German for it

Translation of House	Count
Haus	8000
Haushal	1600
Heimat	200
Gebäude	150
Schale	50

Since haus has the maximum count the word to be replaced is haus.
Hence the output sentence processed from Translation Model will be
"**Ich ging ins kino haus am Sonntag**"

Step III: Reordering
As the output of the translation model is not idealized. In order to preserve the grammatical construct of the sentence reordering needs to be carried out.
The following sentence is the output of the reordering phase "**Am Sonntag ging ich ins kino haus**"

Final Result:
Input sentence: **I went to the cinema house on Sunday.**
Output sentence: **Am Sonntag ging ich ins kino haus.**

V. MOSES EXPERIMENTAL SETUP

A. Installation

Moses is a toolkit which requires a Linux platform. In this section, we describe a step by step installation of Moses on the Ubuntu operating system with version 10.04. Each step is a set of commands which need to be executed in the shell.

i) Install required packages:

```
sudo apt-get install build-essential subversion zlib1g-dev autoconf automake g++ make gawk gzip tcl8.4 tcl8.4-dev tcsh tcl-dev tclx8.4 tclx8.4-dev libtool libboost-dev libboost-graph-dev libboost-iostreams-dev libboost-program-options-dev gcc-4.1 g++-4.1 git-core
```

ii) **Set GCC-4.1 as default compiler**

```
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-4.4 40
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-4.1 30
sudo update-alternatives --config gcc
sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-4.4 40
sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-4.1 30
sudo update-alternatives --config g++
```

iii) **Default folders**

```
sudo mkdir -p /opt/tools/bin
cd /opt/tools
```

iv) Install GIZA++

GIZA++ is a program for aligning words and sequences of words in sentence aligned corpora.

```
sudo wget http://giza-pp.googlecode.com/files/giza-pp-v1.0.5.tar.gz
sudo tar -xzvf giza-pp-v1.0.5.tar.gz
cd /opt/tools/giza-pp
sudo make
cd /opt/tools
sudo cp giza-pp/GIZA++-v2/GIZA++ bin/
sudo cp giza-pp/mkcls-v2/mkcls bin/
sudo cp giza-pp/GIZA++-v2/snt2cooc.out bin/
```

v) **Install SRILM**

SRILM is used for constructing statistical language models (LMs). It is being developed in the SRI Speech Technology and Research Laboratory. Moses depends on SRILM to compile and create LMs and translations.

```
sudo mkdir /opt/tools/srilm
cd /opt/tools/srilm
```

NOTE: SRILM download requires web registration. This can be done on the official SRILM website. You will get a .tgz file which needs to be copied to this directory.

```
sudo tar -xzvf srilm.tgz
sudo rm  srilm.tgz
sudo chmod +w Makefile
sudo gedit Makefile
```

NOTE: Replace the value of the following path variable in the file with the correct path.
SRILM = /opt/tools/srilm

NOTE: Carry out the following steps according to your Ubuntu version.

For Ubuntu 32 Bits:

```
sudo make NO_TCL=1 World
sudo make NO_TCL=1  all
```

For Ubuntu 64 Bits:

```
sudo make NO_TCL=1 MACHINE_TYPE=i686-m64 World
sudo make MACHINE_TYPE=i686-m64 all
```

NOTE: Moses needs an "i686" folder to run SRILM. Perform the following steps for it. Replace VERSION according to your computer.

```
cd /opt/tools/srilm/bin
sudo ln -s i686-VERSION i686
sudo cd /opt/tools/srilm/lib
sudo ln -s i686-VERSION i686
```

vi) Install IRSTLM

The IRSTLM provides the ability to use quantized and disk memory-mapped language model.

```
cd /opt/tools
sudo mkdir irstlm
sudo svn co
https://irstlm.svn.sourceforge.net/svnroot/irstlm/trunk irstlm
cd /opt/tools/irstlm
sudo . regenerate-makefiles.sh
sudo ./configure
sudo make
sudo make install
```

vii) **Install MOSES**

```
cd /opt/tools
sudo git clone https://github.com/moses-smt/mosesdecoder.git
sudo mv mosesdecoder moses
cd moses
```

```
sudo ./regenerate-makefiles.sh
sudo ./configure --with-srilm=/opt/tools/srilm –with-irstlm=/usr/local/irstlmo
sudo make -j 2
```

NOTE: The -j 2 is optional. make -j X where X is number of simultaneous tasks is a speedier option for machines with multiple processors

viii) **Test Moses**

```
cd /opt/
sudo mkdir data
cd data
sudo wget http://www.statmt.org/moses/download/sample-models.tgz
sudo tar -xzvf sample-models.tgz
cd sample-models
sudo su
```

NOTE: Check the lmodel-file parameters. The first parameter should have 0 value for SRILM. So set the first parameter to 0.

```
/opt/tools/moses/moses-cmd/src/moses -f /opt/data/sample-models/phrase-model/moses.ini < /opt/data/sample-models/phrase-model/in > out
```

ix) **Install MOSES SCRIPTS**

```
cd /opt/tools
sudo mkdir moses-scripts
cd /opt/tools/moses/scripts
sudo gedit Makefile
```

NOTE: Replace the values of the following path variables in the file with the correct path.
TARGETDIR?=/opt/tools/moses-scripts
BINDIR?=/opt/tools/bin/

```
sudo make release
```

NOTE: After getting error of check-dependcies.pl go to Makefile in scripts and replace @ with perl

Once make is finished, a folder will be created with the compiled scripts named with the current machine date and time. Now we need to set a SCRIPTS_ROOTDIR environment variable with the path to the scripts. Replace YYYYMMDD-HHMM in the following lines of code according to your machine.

```
sudo gedit /etc/bash.bashrc
export SCRIPTS_ROOTDIR=/opt/tools/moses-scripts/scripts-YYYYMMDD-HHMM
```

x) **Install OTHER SCRIPTS**

```
cd /opt/tools/
sudo wget http://homepages.inf.ed.ac.uk/jschroe1/how-to/scripts.tgz
sudo tar -xzvf scripts.tgz
cd scripts
sudo wget ftp://jaguar.ncsl.nist.gov/mt/resources/mteval-v11b.pl
sudo chmod +x mteval-v11b.pl
```

VI. SUMMARY

Conventional machine translation techniques require large amounts of linguistic knowledge. To overcome this, another approach called statistical machine translation has been proposed in literature. In this paper, we have explained how SMT searches for analogies between the features of two languages from a parallel bilingual corpus and incorporates them in the translation process. Also, the basic mathematics of SMT, related to probability distribution theory, has been discussed. For any SMT system, the corpus is the foundation on which the system is structured. We have described the steps for the acquisition of parallel bilingual corpus and its role in building statistical machine translation systems. In addition, this paper illustrates the architecture of an open-source SMT decoder called Moses, which can incorporate some of the linguistic features in a consistent and flexible framework. The installation steps involved in building the decoder.

A drawback with the statistical translation model is that it presumes the existence of an aligned parallel corpus. The corpus has to be large enough so that the model can derive reliable probabilities from it, and it should be representative enough of the domain it is intended to work for. Inspite of this, this approach has been proved to give superior quality results than its counterparts. This new direction in research opens up many possibilities and issues that require further experimentation.

ACKNOWLEDGMENT

We would like to thank Prof. Meenal Mungi for her valuable time, guidance, inputs, support, suggestions and co- operation during this project. Also, we would like to extend our gratitude towards Prof. R.B. Joshi, HOD Computer Department, for his constant inspiration and encouragement.

REFERENCES

[1] A. Ramanathan, “Statistical Machine Translation”, in *Proc.* 2009 Conference on Empirical Methods in Natural Language Processing,2009

[2] Franz Josef Och, “Minimum Error Rate Training in Statistical Machine Translation”, in *Association for Computational Linguistics*, 2003

[3] Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin, "A Statistical Approach To Machine Translation", Computational Linguistics Volume 16

[4] Peter F Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Vincent J. Della Pietra," The Mathematics of Statistical Machine Translation: Parameter Estimation", IBM T.J. Watson Research Center, Yorktown Heights, NY 10598

[5] Philipp Koehn, "Europarl: A Parallel Corpus for Statistical Machine Translation", Information Sciences Institute University of Southern California

[6] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, "Moses: Open Source Toolkit for Statistical Machine Translation", Proceedings of the ACL 2007 Demo and Poster Sessions

[7] S. Sripirakas, A. R. Weerasinghe, D. L. Herath, "Statistical Machine Translation of Systems for Sinhala-Tamil", Advances in ICT for Emerging Regions (ICTer), 2010 International Conference