PAN Localization Project

RESEARCH REPORT

PHASE 3.2

Final Report on Statistical Machine Translation for Bahasa Indonesia-English and English- Bahasa Indonesia

Agency for the Assessment and Application of Technology

Badan Pengkajian dan Penerapan Teknologi (BPPT)

October 2009

Final Report on
Statistical Machine Translation for Bahasa Indonesia-English (BI-E) and English-Bahasa Indonesia (E-BI)

1. Introduction

In the era of globalization, communication among languages becomes much more important. Computer as a tool to help facilitating communication between different languages has been used more actively. People has been hoping that natural language processing and speech processing, which are branches of artificial intelligence with ICT (Information and Communication Technology), can assist in smoothening the communication among people with different languages. However, especially for Indonesian language, there were only few researches on computational linguistics, natural language processing and speech processing in the past.

Machine translation (MT) is a sub-field of computational linguistics that investigates the use of computer software to translate text or speech from one natural language to another. At its basic level, MT performs simple substitution of words in one natural language for words in another. Using corpus techniques, more complex translations may be attempted, allowing for better handling of differences in linguistic typology, phrase recognition, and translation of idioms, as well as the isolation of anomalies.

Despite its involvement in international projects, basic researches in MT are still underdeveloped. For instance, there is no available corpus (collection of linguistic data; written, spoken, or a mixture of the two) – textual corpus for natural language processing researches or spoken corpus for speech processing researches – which is vital and very basic mean to conduct a research in the field of MT, natural language processing or speech processing. In addition, researchers need basic resources to study the morphology and syntax of Bahasa Indonesia for doing further researches. By using a corpus, one may study the possible structure of sentences (both formal and informal language), words frequency, relation among phrases, etc. In short, information contained in a linguistic corpus is very useful and crucial for doing a research in natural language processing or speech processing. Based on the fact that there is no corpus available and its crucial importance, the first phase of this project is to build large bilingual Indonesian-English corpus, which in turn are used to build ready-to-use modules/systems for the statistical machine translation (SMT) of English to Bahasa Indonesia.

As with other parts of the world, Internet has connected Indonesian user with the rest of the world. Internet has been providing wealthy information on seemingly every thing. However, information in Internet is mostly written in English language. The average English proficiency among Indonesians, particularly those living in rural areas and small towns is very low. For many Indonesian people, writing in English is still an obstacle. It will be very useful if there is a tool such as machine translation system to help translating English into Indonesian texts and vice versa. Statistical machine translation tries to generate translations using statistical methods based on bilingual text corpora. The term parallel corpora are typically used in linguistic circles to refer to texts that are translations of each other. In order to exploit a parallel text, some kind of text alignment, which identifies equivalent text segments (approximately sentences), is a prerequisite for analysis.

The goal of statistical machine translation is to translate a source language sequence into a target language sequence by maximizing the posterior probability of the target sequence given the source sequence. In state-of-the-art translation systems, this posterior probability usually is modeled as a combination of several different models, such as: phrase-based models for both translation directions, lexicon models for translation directions, target language model, phrase and word penalties, etc. Probabilities that describe correspondences between the words in the source language and the words in the target language are learned from a bilingual parallel text corpus and language model probabilities are learned from a monolingual text in the target language. The larger the available training corpus used for translation model, then the better the performance of a translation system.

The benefits of statistical machine translation over traditional paradigms that are most often cited are the following:

- Better use of resources
  There is a great deal of natural language in machine-readable format. Generally, SMT systems are not tailored to any specific pair of languages. Rule-based translation systems require the manual development of linguistic rules, which can be costly, and which often do not generalize to other languages.

- More natural translations
  The ideas behind statistical machine translation come out of information theory Essentially, the document is translated on the probability $p(e|f)$ that a string e in native language (for example, English) is the translation of a string f in foreign language (such as Bahasa Indonesia). Generally, these probabilities are estimated using techniques of parameter estimation.

2. Development Framework

There are two important components in forming machine translation system where both this process is important and each other interconnected component, first is the corpora and secondly is Statistical Machine Translation (SMT). The Final Design Framework of English-Indonesia SMT is given in Figure 1. This development framework differs from the one that we reported in the Research Report on Initial Design SMT Framework (Report No.3 PANL Report BPPT Initial Design Framework SMT.pdf)

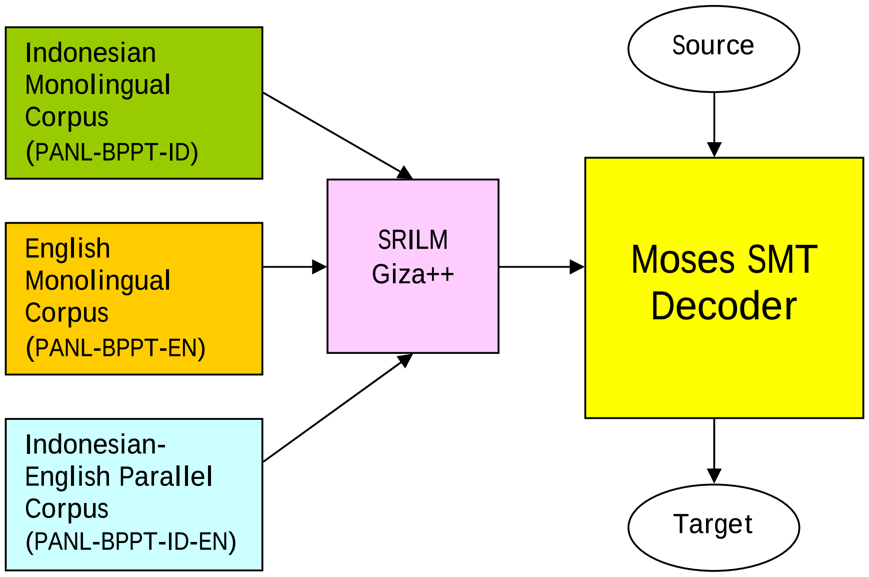
Indonesian Monolingual Corpus (PANL-BPPT-ID)
English Monolingual Corpus (PANL-BPPT-EN)
Indonesian-English Parallel Corpus (PANL-BPPT-ID-EN)
SRILM Giza++
Source
Moses SMT Decoder
Target

Figure 1. Bidirectional Indonesian-English Development Framework

Statistical machine translation (SMT) is a machine translation paradigm where translations are generated on the basis of statistical models whose parameters are derived from the analysis of bilingual text corpora. The statistical approach contrasts with the rule-based approaches to machine translation as well as with example-based machine translation.

The first ideas of statistical machine translation were introduced by Warren Weaver in 1949[1], including the ideas of applying Claude Shannon's information theory. Statistical machine translation was re-introduced in 1991 by researchers at IBM's Thomas J. Watson Research Center[2] and has contributed to the significant resurgence in interest in machine translation in recent years. Nowadays it is by far the most widely-studied machine translation method.

Machine translation (MT) is a sub-field of computational linguistics that investigates the use of computer software to translate text or speech from one natural language to another. At its basic level, MT performs simple substitution of words in one natural language for words in another. Using corpus techniques, more complex translations may be attempted, allowing for better handling of differences in linguistic typology, phrase recognition, and translation of idioms, as well as the isolation of anomalies.

Current machine translation software often allows for customization by domain or profession (such as news) - improving output by limiting the scope of allowable

[1] W. Weaver (1955). Translation (1949). In: *Machine Translation of Languages*, MIT Press, Cambridge, MA.

[2] P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer (1993). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, **19(2)**, 263-311.

substitutions. This technique is particularly effective in domains where formal or formulaic language is used. It follows then that machine translation of government and legal documents more readily produces usable output than conversation or less standardized text.

Improved output quality can also be achieved by human intervention: for example, some systems are able to translate more accurately if the user has unambiguously identified which words in the text are names. With the assistance of these techniques, MT has proven useful as a tool to assist human translators, and in some cases can even produce output that can be used "as is". However, current systems are unable to produce output of the same quality as a human translator, particularly where the text to be translated uses casual language.

Statistical machine translation tries to generate translations using statistical methods based on bilingual text corpora. The term parallel corpora are typically used in linguistic circles to refer to texts that are translations of each other. And the term comparable corpora refer to texts in two languages that are similar in content, but are not translations. In order to exploit a parallel text, some kind of text alignment, which identifies equivalent text segments (approximately sentences), is a prerequisite for analysis.

To produce a good translation as does human translation hence needed a good bilingual text corpus. There is many sources providing various article type to be made by collection of corpus, however require to be conducted election to written article use good structure method. We still find some articles not in good sentence structure, so that to make parallel sentence still needed repairs.

2.1. Corpora

As a language is dynamic and constantly evolving, it is essential that the constructed linguistic resources are based on empirical evidence. To support this, the first phase of the work involves the compilation of corpora, i.e. monolingual collection of electronic texts in Bahasa Indonesia and bilingual collection of parallel English and Indonesian texts. The choice of documents is affected by the intended coverage of the linguistic resources.

- PANL-BPPT-ID Monolingual Indonesian corpus (Bahasa Indonesia)

  The collection is built on 500,000 words corresponding to around 50,000 sentences in Bahasa Indonesia. In order to create the target language model (Indonesia language model) more accurately, it is planned that the PANL-BPPT monolingual corpus will be combined with ANTARA corpus which consist of approximately 2,500,000 sentences. The corpus will contain formal and informal words, and various sentence structures.

- PANL-BPPT-EN Monolingual English corpus (Translation of Indonesian corpus)

  It is planned that we will use the result of translation of PANL-BPPT-ID to develop English monolingual corpus which contains various domain. It will consist of approximately 50,000 sentences.

- PANL-BPPT-ID-EN Parallel corpus (Bahasa Indonesia and English)

  The parallel corpus (PANL-BPPT-ID-EN) is constructed through sentence alignment process of Indonesia and English Monolingual Corpus. The total collection will contain 500,000 words of Indonesian, approximately more than 50,000 sentences for each language, making a total of 100,000 sentences in both languages.

2.2 Statistical Machine Translation Toolkit

The toolkit is a complete out-of-the-box translation system for academic research. It consists of all the components needed to preprocess data, train the language models and the translation models. It also contains tools for tuning these models using minimum error rate training (Och 2003) and evaluating the resulting translations using the BLEU score (Papineni et al. 2002).

Moses uses standard external tools for some of the tasks to avoid duplication, such as GIZA++ (Och and Ney 2003) for word alignments and SRILM for language modeling. Also, since these tasks are often CPU intensive, the toolkit has been designed to work with Sun Grid Engine parallel environment to increase throughput. In order to unify the experimental stages, a utility has been developed to run repeatable experiments. This uses the tools contained in Moses and requires minimal changes to set up and customize.

The toolkit has been hosted and developed under sourceforge.net since inception. Moses has an active research community and has reached over 1000 downloads as of 1st March 2007. The main online presence is at http://www.statmt.org/moses/ where many sources of information about the project can be found. Moses was the subject of this year's Johns Hopkins University Workshop on Machine Translation (Koehn et al. 2006).

The decoder is the core component of Moses. To minimize the learning curve for many researchers, the decoder was developed as a drop-in replacement for Moses, the popular phrase-based decoder. In order for the toolkit to be adopted by the community, and to make it easy for others to contribute to the project, we kept to the following principles when developing the decoder:

- Accessibility
- Easy to Maintain
- Flexibility
- Easy for distributed team development
- Portability

The decoder was originally developed for the phrase model proposed by Marcu and Wong. At that time, only a greedy hill-climbing decoder was available, which was insufficient for our work on noun phrase translation (Koehn, 2003). The decoder implements a beam search and is roughly similar to work by Tillmann and Och. In fact, by reframing Och's alignment template model as a phrase translation model, the decoder is also suitable for his model, as well as other recently proposed phrase models. The phrase-based decoder is developed by employing a beam search algorithm, similar to the one used by Jelinek for speech recognition. The Bahasa Indonesia output sentence is generated left to right in form of hypotheses.

The development objective on our work is to build a statistical machine translation toolkit and make it available to researchers in our PANL community. Therefore, this SMT toolkit would include corpus preparation software, bilingual text training software, and run time decoding software for performing actual translation. All these software components are based on Open Source Software (OSS).

3. Implementation of Bidirectional English-Bahasa Indonesia Statistical Machine Translation

3.1 Hardware Preparation

In developing the SMT we will use a server with the computing capacity to process a big translation table, minimum of 1.3 GHz. The SMT decoder requires memory of 2 GB. We opt to select the latest server technology which is based on dual core or quad core CPU. We use the memory system which can handle error rate correction. The network adapter must have 10/100/1000 Mbps bandwidth access, which is highly important when running parallel processes for servicing multiple client. The configuration is as follows:

Processor (max)	Up to 2 quad-core Intel® Xeon® X5500 series with Intel QuickPath Interconnect (QPI) technology, up to 2.93 GHz and up to 1333 MHz front-side bus
Number of processors (std / max)	1/2
Cache (max)	Up to 8 MB
Memory[1] (max)	1 GB, 2 GB, 4 GB or 8 GB DDR-3 RDIMMs with 16 slots up to 128 GB maximum memory
Expansion slots (I / O)	4 PCI-Express (4x8) Gen 2 slots: 2x8 full-length, full-height; 1x8 half-length, full-height; 1x8 low-profile. 4x8 are convertible to 2x16 via optional risers. Also, 2x PCI-X via optional risers
Disk bays (total / hot-swap)	Up to twelve 2.5" hot-swap Serial Attached SCSI (SAS)/Serial ATA (SATA) HDDs or solid state drives (SSDs)
Maximum internal storage[1,2]	Up to 3.6 TB hot-swap SAS or up to 3.6 TB hot-swap SATA or up to 600 GB hot-swap SSD storage
Network interface	Integrated 2 ports, plus 2 ports optional Gigabit Ethernet
Power supply (std / max)	1/2; 675 W each
Hot-swap components	Power supplies, fan modules, disks
RAID support	Hardware RAID-0, -1, optional RAID-5, -6

3.2 Data Corpus Preparation

In order to prepare the SMT for English-Bahasa Indonesia, we need to train the toolkit using a new language pairs. The need for bilingual texts for training is compulsory for the SMT decoder; hence we use the collected corpus PANL-BPPT (parallel corpus). The description of Corpus files can be found in the Deliverables (provided in a CD ROM) organize into:

1 Corpus 100K Phase 1
   1.1. PANL-BPPT-ID-100K-1.xml
   1.2. PANL-BPPT-EN-100K-1.xml,
   1.3. PANL-BPPT-Project-100K-1-omegaT.tmx

2. Corpus 150K Phase 1
   2.1. PANL-BPPT-ID-150K-1.xml
   2.2. PANL-BPPT-EN-150K-1.xml
   2.3. PANL-BPPT-Project-150K-1-omegaT.tmx

3. Corpus 150K Phase 2
   3.1. PANL-BPPT-ID-150K-2.xml,
   3.2. PANL-BPPT-EN-150K-2.xml,
   3.3. PANL-BPPT-Project-150K-2-omegaT.tmx

4. Corpus 100K Phase 2
   4.1. PANL-BPPT-ID-100K-2.xml
   4.2. PANL-BPPT-EN-100K-2.xml
   4.3. PANL-BPPT-Project-100K-2-omegaT.tmx

3.3 Language Model Preparation

The tool that we use is SRILM (SRI Language Modeling). SRILM is a toolkit for building and applying statistical language models (LMs), primarily for use in speech recognition, statistical tagging and segmentation, and machine translation. It has been under development in the SRI Speech Technology and Research Laboratory since 1995. The toolkit has also greatly benefited from its use and enhancements during the Johns Hopkins University/CLSP summer workshops in 1995, 1996, 1997, and 2002. This tool available in http://www.speech.sri.com/projects/srilm/download.html

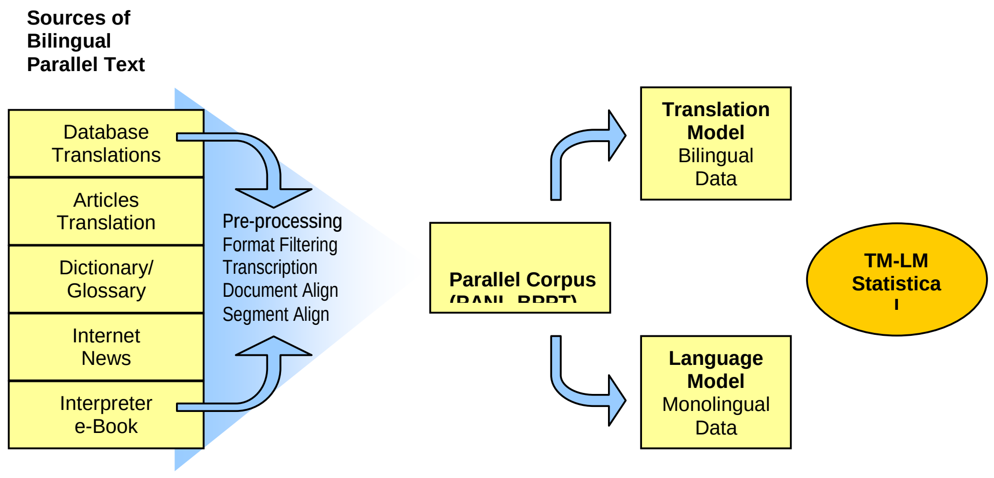

Figure 2. Data Processing of Statistical Model for SMT

Install SRILM in CentOS 64 bit and create folder srilm

Instalation package gcc for CentOS 64 bit, the syntax in the command line are

```
# yum install gcc4-c++-4.1.2-44.EL4.x86_64.rpm gawk gzip csh

# mkdir srilm
# cd srilm
```

Extrack SRILM

```
$ tar -xzvf srilm.tgz
```

Edit Makefile

```
$ vi Makefile
###
# SRILM = /usr/.../.../.../devel
MACHINE_TYPE
###
SRILM = /home/.../.../Sources/srilm
MACHINE_TYPE = i686-m64
###
```

Move to common folder

```
cd common
```

edit Makefile.machine.i686-m64

```
###
CC = /usr/bin/gcc $(GCC_FLAGS)
CXX = /usr/bin/g++ $(GCC_FLAGS) -DINSTANTIATE_TEMPELATES
.
.
.
TCL_INCLUDE =
TCL_LIBRARY =
NO_TCL = 1
###
```

Move to srilm folder

```
cd ../srilm
```

Type the command and run it

```
make World
```

Type the command
add SRILM to global PATH variable.

```
export PATH=/home/.../.../Sources/srilm/bin/i686-m64:/home/.../.../Sources/srilm/bin:$PATH
```

Move to folder test

```
cd test
```

Type the command

```
make all
```

And we are done with SRILM!

Install SRILM in Ubuntu 9.04 and create folder srilm

Instalation package gcc for Ubuntu 9.04, the syntax in the command line are

```
$ sudo apt-get install g++ make gawk gzip tcl8.5 tcl8.5-dev csh

$ mkdir srilm
$ cd srilm
```

Extrack SRILM

```
$ tar -xzvf srilm.tgz
$ edit Makefile
```

Edit Makefile

```
###
# SRILM = /usr/.../.../.../devel
MACHINE_TYPE
###
SRILM = /home/-path-to-/srilm
MACHINE_TYPE = $ (shell $(SRILM)/sbin/machine-type)
###
```

Move to common folder

```
cd common
```

edit Makefile.machine.i686

```
###
CC = /usr/bin/gcc $(GCC_FLAGS)
CXX = /usr/bin/g++ $(GCC_FLAGS) -DINSTANTIATE_TEMPELATES
.
.
.
TCL_INCLUDE = -I/usr/include/tcl8.5/
TCL_LIBRARY = /usr/lib/libtcl8.5.so
###

Move to srilm folder
$ cd ../srilm
$ make World
```

Type the command
add SRILM to global PATH variable.

```
export
PATH=/home/.../.../sources/srilm/bin/i686:/home/.../.../Sources/srilm/bin:$PATH
```

Move to folder test

```
cd test
make all
```

And we are done with SRILM!

Install Giza++

GIZA++ is a statical machine translation toolkit that is used to compute word alignments between two sentence aligned corpora. This package also contains the source for the mkcls tool which generates the word classes necessary for training some of the alignment models.

mkcls is a tool to train word classes by using a maximum-likelihood-criterion. The resulting word classes are especially suited for language models or statistical translation models. (http://code.google.com/p/giza-pp/)

Installation Giza on CentOS 64 bit

Current version will compiled with newer `gcc4-c++-4.1.2` compilers which are standard on modern computer system,

Download *giza-pp-v1.0.3.tar.gz* and extract it

```
# tar –xzvf giza-pp-v1.0.3.tar.gz
# cd GIZA++-v2/
```

Edit Makefile

```
###
CXX = g++ ---> CXX = g++4
```

Then erase -DBINARY_SEARCH_FOR_TTABLE

Move to folder gizapp

```
# cd ../gizapp
# make
```

Copy GIZA++ and mkcls to a bin location for Moses Scripts in /home/.../.../Sources/

Create folder bin

```
# mkdir -p bin
# cp GIZA++-v2/GIZA++ bin/
# cp GIZA++-v2/snt2cooc.out bin/
# cp mkcls-v2/mkcls bin/
```

Both GIZA++ and mkcls will be called by Moses training scripts.
And we are done with Giza++!

Installation Giza on Ubuntu 9.04
Current version will compiled with newer `g++-4.x` compilers which are standard on modern computer system,

Move to folder Giza++-V2

```
$ cd GIZA++-v2/
```

Edit Makefile

```
###
CXX = g++ ---> CXX = g++-4.1
then errase -DBINARY_SEARCH_FOR_TTABLE
###
```

Move to folder gizapp

```
$ cd ../gizapp
$ make
```

Copy GIZA++ and mkcls to a bin location for Moses Scripts in /home/.../.../Sources/

```
$ mkdir -p bin
cp GIZA++-v2/GIZA++ bin/
cp GIZA++-v2/snt2cooc.out bin/
cp mkcls-v2/mkcls bin/
```

Both GIZA++ and mkcls will be called by Moses training scripts and we are done with Giza++!

In this experiment, we use SRILM to functions as follows (see Figure 3):

- Produce the n-gram of a corpus
- Train the language model from the n-gram
- Calculate the perplexity of data test using the trained language model

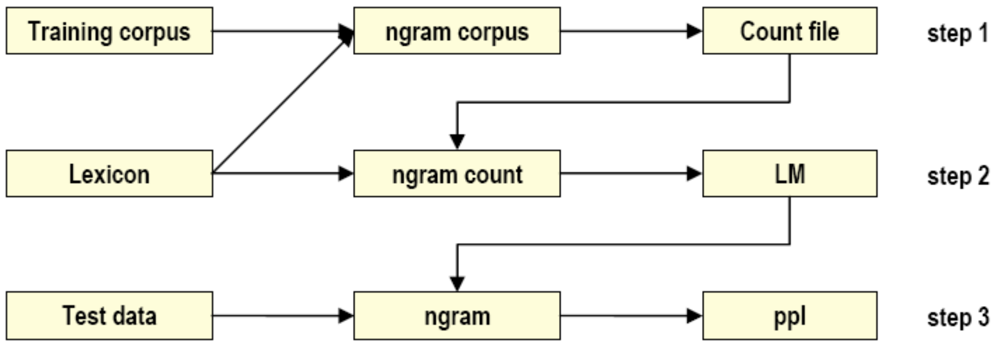

Figure 3. Step-by Step Usage of SRILM

Build Language Model
The most important command of the SRILM toolkit is the ngram-count tool which counts ngrams and estimates language models. There exist several command line switches to fine-tune the resulting language model. A sorted 4-gram language model from a given English corpus in en.corpus can be created using the following command: (Christian Federman, 2007)

Type the command
```
$ /path-to-srilm/bin/i686-m64/ngram-count -order 4 -interpolate -kndiscount -
text working-dir/lm/file –lm working-dir/lm/file.lm
```

The command line switches are explained below:

- -order n sets the maximal order (or length) of n-grams to count. This also determines the order of the language model.
- -interpolate causes the discounted n-gram probability estimates at the specified order n to be interpolated with estimates of lower order.
- -kndiscount activates Chen and Goodman's modified Kneser-Ney discounting for ngrams.
- -text specifies the source file from which the language model data is estimated. This file should contain one sentence per line, empty lines are ignored.
- -lm specifies the target file to which the language model data is written.

Here is Language Model with SRILM toolkit 4-gram

```
\data\
ngram 1=29682
ngram 2=232779
ngram 3=47994
ngram 4=26847


-5.065517       considerations   -0.1159884
-5.239327       considerd-0.1159884
-3.804326       considered       -0.2757038
-3.92718        considering      -0.2303499
-4.269085       considers -0.1523835
-4.269085       consist   -0.7713577
-4.204758       consisted -0.7038553
```

```
-4.964279       consistency     -0.1159884
-4.344625       consistent-0.3910371
-4.819496       consistently    -0.1159884
-4.175847       consisting      -1.047205
-3.828781       consists   -1.201482
-5.065517       console   -0.1159884
-5.239327       consoles  -0.1159884
-5.239327       consolidate     -0.1159884
-4.624427       consolidated    -0.1740036
-5.065517       consolidating   -0.1159884
-4.43613        consortium      -0.1541466
-5.239327       consorzio       -0.1159884
-4.964279       conspiracies    -0.1159884


\2-grams:
-3.15486        many component
-2.601195       many computer
-3.190189       many conglomerates
-3.198194       many considerations
-3.184963       many constraints
-3.193709       many continents
-3.193709       many cooperating
-3.199374       many copyrights
-1.748011       many countries  -0.03104825
-3.181513       many critics
-3.199374       many crn
.
.
.

-3.198194       many exemptions


\3-grams:
-1.727292       points to eat
-0.370703       points to rp    -0.03852752
-1.336449       points to the
-1.271185       possible to be
-1.501909       possible to change
-1.471183       possible to produce
-1.162108       potential to provide
-0.9504622      power to the
-0.4413998      predators to insects
-0.8592629      predicted to be -0.03246684
.
.
.

-0.6168181      promise to respect  -0.1524059


\4-grams:
-1.885318       in the world and
-2.22455        in the world at
-1.596555       in the world crude
-2.102185       in the world economy
-2.240158       in the world have
-1.809982       in the world in
-1.228551       in the world market
.
.
.
-0.4179575      including the world bank
```

3.4 Translation Model Preparation

In the following, we will describe step by step procedures to install **Moses Support Libraries in CentOS-5 64 Bit and Ubuntu 9.04**

Instalation extra package for CentOS 64 bit, the syntax in the command line are

```
# yum install subversion automake autoconf texinfo zlib1g zlib1g-dev zlib-bin
zlibc
```

Instalation extra package for Ubuntu 9.04, the syntax in the command line are

```
$ sudo apt-get install subversion automake autoconf texinfo zlib1g zlib1g-dev
zlib-bin zlibc
```

Install MOSES

Moses is a statistical machine translation system which allows to train translation models for any given language pair for which a parallel corpus, i.e. a collection of translated texts, exists. The Moses decoder works using a beam search [Koehn, 2004a] algorithm to determine the best translation for a given input. Translation is phrase-based [Koehn et al., 2003] and allows words to have a factored representation. Moses has been designed by a team headed by Philipp Koehn. (Christian Federman, 2007)

Installation Moses on Centos 64 bit

The Moses source code can be obtained from the project website or the SourceForge Subversion repository. Check out the Moses code via Subversion:

```
# yum install subversion
# svn co https://mosesdecoder.svn.sourceforge.net/svnroot/mosesdecoder/trunk
```

Assuming that the Moses source code is available in the $Source folder
Compile trunk

```
# cd trunk
# ./regenerate-makefiles.sh
```

Choose the preferred LM toolkit. To do that use the parameter either §with-srilm.

```
# ./configure --with-srilm=/home/.../.../Source/srilm
```

Edit Makefile in trunk, trunk/moses/src/, trunk/moses-cmd/src, trunk/misc, and trunk/mert

```
#
LDFLAGS=/home/../../Sources/srilm/bin/i686

-->  LDFLAGS=/home/../../Sources/srilm/bin/i686-m64
#
```

Type the command

```
# make -j 4
```

At this stage, Compilation has been completed.

Installation Moses on Ubuntu 9.04

The Moses source code can be obtained from the project website or the SourceForge Subversion repository. Check out the Moses code via Subversion:

```
$ sudo apt-get install subversion
$ svn co https://mosesdecoder.svn.sourceforge.net/svnroot/mosesdecoder/trunk
```

Assuming that the Moses source code is available in the $Source folder
Compile trunk

```
$ cd trunk
$ ./regenerate-makefiles.sh
```

Choose the preferred LM toolkit. To do that use the parameter either §with-srilm.

```
$ ./configure –with-srilm=/home/.../.../Source/srilm
```

Type the command

```
$ make -j 4
```

At this stage, Compilation has been completed.

Install Moses Scripts

Compile Moses Scripts
The support scripts used by Moses are "released" by a Makefile which edits their paths to match your local environment. First, you need to edit the Makefile definition of two variables:

Create folder in /home/.../.../Sources/

```
$ mkdir moses-scripts
```

Edit trunk/scripts/Makefile

```
###
TARGETDIR=/home/.../.../Sources/moses-scripts
BINDIR=/home/.../.../Sources/bin
###
```

Move to folder trunk/scripts/

```
$ cd trunk/scripts/
$ make release
```

This will create a folder named moses-scripts/scripts-YYYYMMDD-HHMM with released versions of all the scripts. You will call these versions when training/tuning Moses. Moses scripts also require a SCRIPTS_ROOTDIR environment variable to be set. The output of make release should indicate this.

Type the command

```
export SCRIPTS_ROOTDIR=/home/.../.../Sources/moses-scripts/scripts-YYYYMMDD-HHMM
```

Additional requirements:
Moses requires some additional tools for the training and evaluation processes. For training, a tokenizer, and a lowercaser are necessary. These tools can be obtained from the website of the 2007 ACL Workshop on Statistical Machine Translation at: http://www.statmt.org/wmt07/scripts.tgz (Christian Federman, 2007)

Download scripts.tgz and extract them:
```
S tar xzf scripts.tgz
```

These scripts include:

- ☞ Tokenizer scripts/tokenizer.perl
- ☞ Lowercaser scripts/lowercase.perl

Prepare data
The first step is to prepare the parallel corpus data. It has to be tokenized, lowercased and sentences which would be too long to handle (and their correspondences in the other language) have to be removed from the corpus.

Tokenize training data
First we split out most punctuation from words. Special cases where splits do not occur are documented in the code.

```
$ mkdir -p working-dir/corpus
$ scripts/tokenizer.perl -l id < working-dir/corpus/file.id > working-
dir/corpus/file.tok.id
$ scripts/tokenizer.perl -l en < working-dir/corpus/file.en > working-
dir/corpus/file.tok.en
```

Filter out long sentences
```
$ bin/moses-scripts/scripts-YYYYMMDD-HHMM/training/clean-corpus-n.perl working-
dir/corpus/file.tok id en working-dir/corpus/file.clean 1 50
```

Lowercase training data
```
$ scripts/lowercase.perl < working-dir/corpus/file.clean.id > working-
dir/corpus/file.lowercased.id
$ scripts/lowercase.perl < working-dir/corpus/file.clean.en > working-
dir/corpus/file.lowercased.en
```

Train Model
Once the corpus data is prepared, the actual training process can be started. translation model training is done using:
Run training script: (enid → -f en -e id)
```
$ bin/moses-scripts/scripts-YYYYMMDD-HHMM/training/train-factored-phrase-
model.perl -scripts-root-dir bin/moses-scripts/scripts-YYYYMMDD-HHMM -root-dir
enid -corpus working-dir/corpus/file.lowercased -f id -e en -alignment grow-
diag-final-and -reordering-smooth msd-bidirectional-fe –lm 0:4:working-
dir/lm/file.id.lm:0
```

Run training script: (iden → -e en -f id)
```
$ bin/moses-scripts/scripts-YYYYMMDD-HHMM/training/train-factored-phrase-
model.perl -scripts-root-dir bin/moses-scripts/scripts-YYYYMMDD-HHMM -root-dir
iden -corpus working-dir/corpus/file.lowercased  -e id -f en -alignment grow-
diag-final-and -reordering-smooth msd-bidirectional-fe -lm 0:4:working-
dir/lm/file.en.lm:0
```

The command line switches are explained below:

- -corpus file corpus -e
- -e english
- -f foreign language
- -alignment word-to-word alignment
- grow-diag-final-and number of word alignment
- -reordering-smooth msd-bidirectional-fe for reordering alignment probability
- -lm Language Model parameter

The training process takes place in nine steps, all of them executed by the script:

1. Prepared data. Training data has to be provided sentence aligned in two file (etc file.en and file.id).
2. Run Giza++. Giza++ is used to compute word alignments between two sentence aligned corpora and mkcls is to generates the word classes.
3. Align words. To establish word alignments based on the two GIZA++ alignments.
4. Get lexical translation table. To estimate a maximum likelihood lexical translation table.
5. Extract Phrases. All phrases are dumped into one big file.
6. Score Phrases. To estimate the phrases translation probability.
7. Building reordering model.
8. Build generation model.
9. Create configuration file. Create moses.ini

The training process takes a lot of time and memory to complete.

Test the moses.ini

```
$ echo 'former world number one' | ../../trunk/moses-cmd/src/moses -f
model/moses.ini > out


Moses (built on Aug 29 2006)
a beam search decoder for phrase-based statistical machine translation models
written by Hieu Hoang, with contributions by Nicola Bertoldi, Ondrej Bojar,
Chris Callison-Burch, Alexandra Constantin, Brooke Cowan, Chris Dyer, Marcello
Federico, Evan Herbst, Philipp Koehn, Christine Moran, Wade Shen, Richard Zens.
(c) 2006 University of Edinburgh, Scotland
command: ../../moses-cmd/src/moses -f moses.ini
Defined parameters (per moses.ini or switch):
config: moses.ini
input-factors: 0
lmodel-file: 0 0 4 ../lm/file.id.lm
mapping: T 0
ttable-file: 0 0 1 phrase-table
ttable-limit: 10
weight-d: 1
weight-l: 1
weight-t: 1
weight-w: 0
Start loading LanguageModel ../lm/file.id.lm : [0.00] seconds
Finished loading LanguageModels : [3.00] seconds
IO from STDOUT/STDIN
Start loading PhraseTable phrase-table : [3.00] seconds
Finished loading phrase tables : [3.00] seconds
```

```
Created input-output object : [3.00] seconds
End. : [3.00] seconds

$ vi out
mantan nomor satu dunia
```

4. Evaluation and Related Issues to SMT Design

4.1 Evaluation of Translation Quality

We believe more work on evaluation is called for. There was insufficient time during the Phase 1.2 to fully evaluate the quality of output from the Toolkit prototype using Moses Decoder, but we record some of our thoughts here.

Most MT evaluation regimes involve collecting human judgments over many sentences and these regimes are expensive for individual researchers. In the commercial world, the market provides a constant evaluation of translation software, albeit one that seems currently driven by factors other than output text quality. In the final analysis, there is no measure that can substitute for user satisfaction. But this leaves the individual researcher in a bind.

It is ideal that human subjects fluent in both the source language and the target language evaluate the material. Monolingual users of the target language alone are poorly prepared to evaluate the content of the translation, despite reference translations. This point is proven with the DARPA evaluations of 1000 when the quality of human translations of variable quality, were also used for evaluation in an exam that asked evaluators to answer questions on the content of a translated passage. A monolingual evaluator is best suited for evaluating the fluency of the output, and not the content, unless the content can really be assured, such as in treaties and other legal documents where the translator has a very high incentive for ensuring that precise translations are produced.

Another concern that arises relates to the use of native speakers of the source language and their capacity to determine the fluency of output in the target language, as well as possible biases which allow them to find meaning in "word hash" output from MT that a monolingual speaker might missing. In defining the scale for evaluation, we aim to address this concern. Since the system is trained on texts with unique properties in terms of lexicons used translation styles involved, and so on. We concluded that it was reasonable to draw evaluation texts from the pre-training corpus. However, we strongly emphasize that in no way is the system to be trained on texts that will later be used for evaluation.

The speech recognition community has largely focused on word -error recognition rate, in the belief that, this figure will ultimately correlate with user satisfaction. And it is easy to measure. In finding text for evaluation, there were several concerns length, context, and novelty. Because of limits on the computational complexity of decoding and training the corpora was stripped of sentences of length above a certain bound, originally 10 tokens. This processed corpus was then used for extracting sentences for evaluation. In addition, for the length of the entire evaluation, we suggest a test-bed of 100 sentences. As 100 sentences seem like a reasonably small amount of work for evaluators per evaluation, we decided upon 1000 test sentences from the outset (out of 50,000 sentences in PANL-BPPT corpus).

Evaluation is done using the BLEU scoring tool. Assuming a reference translation evaluation.en exists, we can evaluate the translation quality of output.en which was translated from evaluation.de as follows:

To produce output evaluation from Moses ;
```
$ ./moses -f ./moses.ini <file_name.lowercase.id>
file_name.id.out
$ ./moses -f ./moses.ini <file_name.lowercase.en>
file_name.en.out
```

To produce output evaluation from MultiBLEU
```
$ multi-BLEU.perl file_name.lowercase.id < file_name.id.out >&
BLEU
$ multi-BLEU.perl file_name.lowercase.en < file_name.en.out >&
BLEU
```

4.2 Training Set Size vs. Translation Quality

One question we had about statistical machine translation was how much data do we need? Are sentence pairs enough? If you go from one million sentence pairs to two million, how much improvement will you see? We imagined that a lot of data is useful, if you've never seen the phrase "real estate" before in your parallel corpus, then you probably aren't going to translate it correctly. It is mandatory in the machine learning community to plot data size on the x axis vs. performance on the y axis, depicting a learning curve of SMT.

This is particularly important in natural language processing, where we can frequently purchase more data. The learning curve leads naturally to a second question what to plot on the y axis. Machine translation is notoriously difficult to evaluate. It shares this notoriety with other tasks that involve generating human language as opposed to interpreting it. It is possible to evaluate a speech recognizer by counting word errors, not so with a speech synthesizer. Likewise, it is easier to say whether a language interpretation system got the right syntactic structure than to say whether a generation system produced a good syntactic structure.

Machine translation involves both interpretation and generation. Following many of the evaluation regimes that have been proposed for MT, we decided to go with either a simple scoring mechanism or a simple relative ranking of translations from different systems configurations. Researchers can try discard and adopt many new ideas without involving human subjects. They can also compare results on common test data sets. As mentioned above, even this is difficult to do in translation. Interestingly, many speech researchers find it convenient to evaluate their ideas with respect to perplexity: a measure of how well a statistical model fits or explains the data. For example, in language modeling, the goal is to produce a program that assigns a probability P(e) to every string of words e*. All these probabilities sum to 1 so there is only so much to go around. It is possible to ask how good a language model is without making reference to word error rate or any other task level measure. One simply asks for the particular number P(e) that a particular instantiated model assigns to a particular text. If the text is good English, we expect P(e) to be high if the text is bad English, we expect P(e) to be low. If we observe a language model which is assigning probabilities the other way around, then it probably isn't a very good language model.

It is reasonable to ask for the P(e) that a model assigns to the text it was trained on. In this case, a memorizing program would do very well, by assigning P(e|f). However, this program would by definition assign P(e|f) to every other text e* and this will lead to a very poor word error rate. Therefore a more reasonable evaluation is test set perplexity, which is related to the probability that the model assigns to a previously unseen test English text. The language model must lay bets on all kinds of strings. If it concentrates its bets on certain subset of strings, then it must hope that when the previously unseen text is revealed it is to be found in that subset. Because all probabilities sum to 1, increasing our bet on one string necessarily means decreasing our bet on some other string.

It is typical in language modeling to bet at least something on every conceivable string e. If we accidentally bet nothing on e, then our P(e) would be zero, and our perplexity would be infinite. So, if a language model uses previously observed word pair frequencies in constructing a probability score P(e) for a new string, it will typically smooth these frequencies so as to accept a string that contains novel word pairs. We evaluated the translation quality of the system using the BLEU metric. We compared our system to Moses decoder, a leading SMT decoder.

The result of BLEU score is presented in Table 1, Figure 4, and Figure 5.

Table 1. BLEU score for English-to-Indonesia and Indonesia-to-English.

Word	Eng-Ina	Ina-Eng
60000	0.775	0.802
120000	0.802	0.802
180000	0.804	0.806
240000	0.824	0.806
280000	0.838	0.807
320000	0.843	0.812
380000	0.927	0.908
400000	0.928	0.918
500000	0.938	0.926

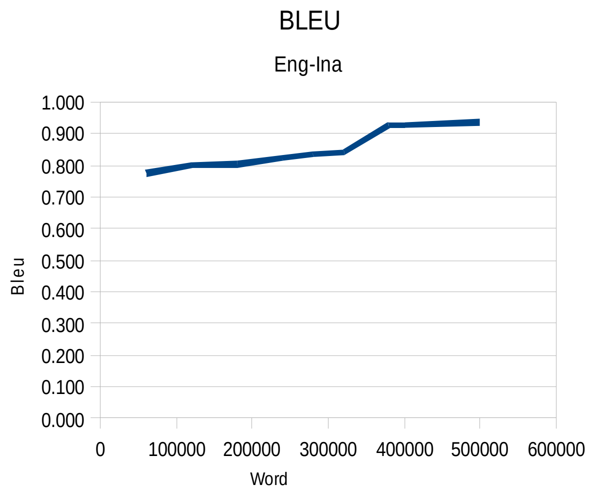

Figure.4 Graph of BLEU score for English-to-Indonesia

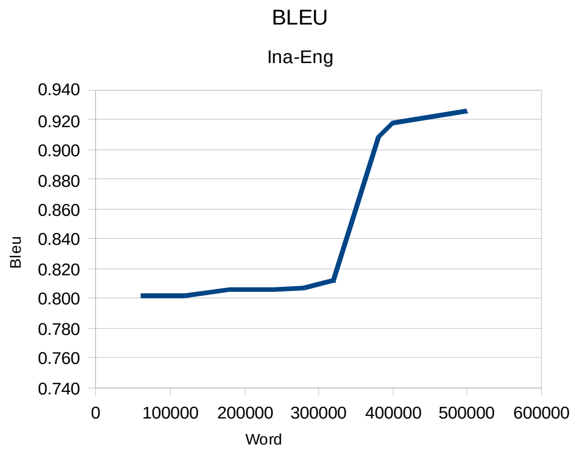

Figure 5. Graph of BLEU score for Indonesia-to-English

We computed all system on the 500.000 word into 9 block of word Indonesia-to-English and English-to-Indonesia, and computed the BLEU Scored on these blocks individually. Statistic of the data sets is shown in Table 1.

The language model used for all models (include decoding models and system combination models) is a 4-gram model trained into 9 samples. We thus have 9 samples of BLEU Score for each system. The first observation showed significantly increasing BLEU score. In Figure 4 and Figure 5 show how BLEU score was improved with increasing size of training corpus. Each curve on the graph represent corpus size, for example, 400.000 word corpus have BLEU score of 0,918 and for the 500.000 word corpus, BLEU score rise to 0,926. This concludes that the BLEU score have improve by improving the translation model. Figure 5 shows the effect of maximal word count on BLEU scores. With the increase of maximal word count, the BLEU score increased dramatically. Finally, the performance of Statistical Machine Translation had to be improving by using larger size of training data. If the data is huge, the performance could better. A higher BLEU score indicates better translation.

5. Conclusion and Future Work

We have reported in this research work the development of statistical MT system for Bahasa Indonesia – English. There is significant improvement in the output quality of English-Indonesia translations compare to our previous work on symbolic approach.

Using 500K words training corpus, we were able to achieve 92.1% translation quality for English to Bahasa Indonesia (easier task than other direction). Many interesting research directions remain open. We can use different variation distributions other than the n-gram model. Interpolation with other models is also interesting.

We analyzed currently used approach to automatic MT evaluation, the BLEU score. Our show analysis that BLEU score have some flaws; one needs to understand these problems to correctly interpret the reported BLEU scores. We also described a method to calculate the statistical significance for BLEU using another Score metrics. These tests will give us the confidence interval for the BLEU scores. In the future action is suggested by the smaller sentences are much more fluent in translation compared to medium length and long sentences.

The Future Work should be in the line of combining Symbolic-Statistical approaches to increase the quality and accuracy of bidirectional Bahasa Indonesia – English Machine translation. In order to achieve better system, the machine translation system will consist of the above modules. Statistical method or combination of statistical and example-based method (using the parallel corpus) will be employed. The hybrid machine translation system will consist of the following modules:

- Morphological analyzer/POS Tagger: the derivational morphology of Bahasa Indonesia, particularly that of verbal morphology, is quite complex, and therefore requires careful analysis. Existing Indonesian morphological literature (Alwi, et. al. 2003) will be surveyed and compared with the empirical evidence as found in the corpus. The morphological analysis will be built using statistical methods such as n-gram or HMM (Manning and Schütze 1999).
- Shallow parser/ Syntactic parser: completion of the morphological analysis will enable the development of phrase parser. This parser will decide noun phrases, verb phrases, adjective phrases, etc. in a sentence. This parser will be built using statistical methods also. Furthermore, a robust syntactic parser can be build using output of the shallow parser to decide syntactical structure of a sentence, i.e., which part of a sentence is the subject, predicate, object.
- Phrase Reordering System: this module will perform transformation of phrase structure from Indonesian DM to English MD, to enable correct translation of

noun phrase and adjective phrase. This module will prepare the input sentence or files prior of translation process, to enable better translation quality

- Machine Translation: the statistical MT system will contain 2 modules, each for English-Indonesian and Indonesian-English translation modules.
- Generation system: this module will produce target sentences (Indonesian or English) based on an intermediate representation created by the statistical MT.

References

1. Riza, Hammam, Budiono and Chairil Hakim. Resource Report: Building Parallel Text for Multi-Domain Translation System, 7th Workshop on Asian Language Resource, Proceedings of ACL-IJCNLP 2009, Singapore, 6-7 August 2009.
2. Jürgen Handke, The Structure of the Lexicon: Human Versus Machine, Walter de Gruyter, 1995
3. Sonja Nießen and Hermann Ney. 2004. Statistical machine translation with scarce resources using morpho-syntactic information. Computational Linguistics, 30(2):181–204, June.
4. Brown et al, 1993 "The Mathematics of Statistical Machine Translation: Parameter Estimation", Computational Linguistics, 19(2).
5. Brown, P.F., J Cocke, S A Della Pietra, V J Della Pietra, F Jelinek, J D Lafferty, R L Mercer, and P S Roosin, P.S. (1990) A statistical approach to machine translation. Computational Linguistics, 16(2):29–85.
6. Chris Callison-Burch and Miles Osborne. 2003. Cotraining for statistical machine translation. In Proc. Of the 6th Annual CLUK Research Colloquium, Edinburgh, UK, January.
7. Knight, 1997 "Automating Knowledge Acquisi-tion for Machine Translation", K. Knight, AI Magazine, 18(4).
8. Y. Al-Onaizan, U. Germann, U. Hermjakob, K. Knight, P. Koehn, D. Marcu, K. Yamada, "Translating with Scarce Resources", AAAI-2000.
9. B. Pang, K. Knight, and D. Marcu. "Syntax-based Alignment of Multiple Translations: Extracting Paraphrases and Generating New Sentences," NAACL-HLT-2003.
10. K. Papineni, S. Roukos, T. Ward, J. Henderson, F. Reeder, "Corpus-based Comprehen-sive and Disagnostic MT Evaluation: Initial Ara-bic, Chinese, French, and Spanish Results", NAACL-HLT-2002.
11. Wikipedia SMT, http://en.wikipedia.org/wiki/, retrieved August 08
12. Aston, G. and Burnard, L. The BNC Handbook Edinburgh: Edinburgh University Press., 1998
13. Serge Sharof "In the garden and in the jungle: comparing genres in the BNC and Internet, http://corpus.leeds.ac.uk/serge/webgenres/colloquium/, Sept 2008

14. Garside, R., Leech, G., and McEnery, T. Corpus annotation: linguistic information from computer text corpora, Harlow: Addison-Wesley Longman, 1997.

15. S. Vogel, H. Ney and C. Tillmann. 1996. HMM-based Word Alignment in Statistical Translation. In COLING '96: The 16th International Conference on Computational Linguistics, pp. 836-841, Copenhagen, Denmark

16. P. Koehn, F.J. Och, and D. Marcu (2003). Statistical phrase based translation. In Proceedings of the Joint Conference on Human Language Technologies and the Annual Meeting of the North American Chapter of the Association of Computational Linguistics (HLT/NAACL).

17. P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, E. Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. ACL 2007, Demonstration Session, Prague, Czech Republic

18. W. J. Hutchins and H. Somers. (1992). An Introduction to Machine Translation, 18.3:322. ISBN 0-12-36280-X

19. Sharon Goldwater and David McClosky. 2005. Improving stastistical machine translation through morphological analysis. In Proceeding of the Conf. on Empirical Methods for Natural Language Processing (EMNLP), Vancouver, Canada, October.

20. Adam Lopez and Philip Resnik. 2005. Improved HMM alignment for languages with scarce resources. In 43rd Annual Meeting of the Assoc. for Computational Linguistics: Proc. Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond, pages 83–86, Ann Arbor, MI, June.

21. W. Weaver (1955). Translation (1949). In: Machine Translation of Languages, MIT Press, Cambridge, MA.

22. Moses decoder http://www.statmt.org/moses/

23. Federman, Christian 2007. Very Large Language Model for Machine Translation, Saarland University.

24. Arnold, D.J., Lorna Balkan, Siety Meijer, R.Lee Humphreys and Louisa Sadler (1994). *Machine Translation: an Introductory Guide*, Blackwells-NCC, London.

25. Beesley K. and Karttunen, L. Finite-state non-concatenative morphotactics. In Proceedings of the Fifth Workshop of the ACL Special Interest Group in Computational Phonology (SIGPHON). 2000

26. Brill, E. (1992). A Simple Rule-Based Part of Speech Tagger, Proceedings of the Third Conference on Applied Computational Linguistics, ACL.

27. Manning, Christopher D. and Schütze, Hinrich. Foundations of Statistical Natural Language Processing. Sixth edition. The MIT Press: Cambridge, USA, 2003

28. Merialdo, B. (1994). Tagging English Text with a Probabilistic Model, Computational Linguistic, Vol. 20 No.2, and MIT Press.

Appendix A

Sample Translation Result

http://translator.iptek.net.id/PANL

English	Bahasa Indonesia
Abbas and Haniyeh had originally been due to meet on Saturday but the meeting was put back to Sunday	Abbas dan Haniya mulanya telah dijadwalkan bertemu Sabtu tapi pertemuan itu diundur sampai Ahad
There is a progressive European position towards dealing with the Palestinian unity government	Ada kemajuan sikap Eropa ke arah kesepakatan dengan pemerintah persatuan Palestina
South Africa host the 2010 event	Afrika Selatan akan menjadi tuan rumah Piala Dunia 2010
The second ism sees that backwardness is caused by existence of inequality and gap among countries	Aliran kedua melihat bahwa keterbelakangan itu disebabkan adanya ketidakadilan dan kesenjangan antar negara
The above opinion can be analysis from two sides i e consumer and producer sides	Anggapan di atas dapat kita telaah dari dua sisi yaitu sisi konsumen dan produsen
Asia has long been blighted by soccer corruption scandals	Asia sudah lama dirusak oleh skandal korupsi sepak bola
Several assessments and theories have been produced by economists as solution of inflation problem	Berbagai kajian dan teoripun telah banyak dihasilkan oleh para ekonom sebagai solusi dari persoalan inflasi
In term of income level widening gap occurs	Dalam hal tingkat pendapatan terjadi jurang kesenjangan yang makin lebar
In national context Indonesian awaken was initiated on May 28 1908	Dalam konteks nasional kebangkitan Indonesia di cetuskan pada 28 Mei 1908
Can be imagined how more than 300 years occupation at that time had castrated frame of thinking and mentality of Indonesian people	Dapat dibayangkan betapa lebih 300 tahun sudah penjajahan saat itu mengebiri pola fikir dan mental rakyat Indonesia
From what FIFA has said they have to let him go Batista said in a television interview	Dari apa yang dikatakan FIFA mereka harus membolehkan dia pergi kata Batista dalam wawancara televisi