PAN Localization Project

RESEARCH REPORT

PHASE 2.1

Final Design Report on Statistical Machine Translation Framework

Agency for the Assessment and Application of Technology
Badan Pengkajian dan Penerapan Teknologi (BPPT)
December 2008

Final Design Report on
Statistical Machine Translation Framework

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)

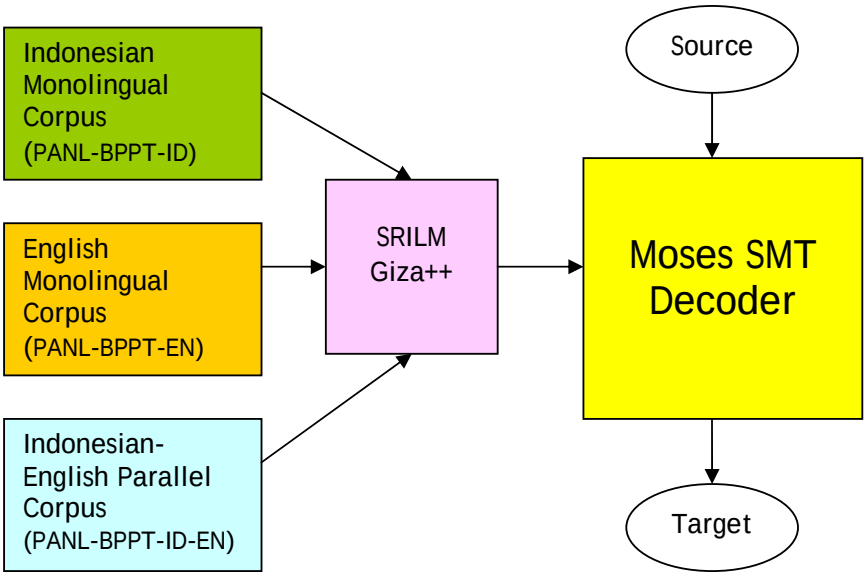

Figure 1. Final Design of Indonesian-English Development Framework

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 acurately, 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

In order to experiment the feasibility of statistical MT for English- Indonesian, we build a prototype of SMT Toolkit using the available Open Source Package. We use SRILM to build the n-gram language model and translation model, experiment with PHARAOH in the initial design stage and subsequently use MOSES (Koehn, 2007) as a beam search decoder.

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).

2.3 Future Systems: Symbolic-Statistical Machine translation

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 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 literatures 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.

- 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

- Generation system:
  This module will produce target sentences (Indonesian or English) based on an intermediate representation created by the statistical MT.

Additional conditional reordering models may be built. These are conditioned on specified factors (in the source and target language), and learn different reordering probabilities for each phrase pair (or just the foreign phrase). Possible configurations are

1. MSD vs. Monotonicity. MSD models consider three different orientation types: monotone, swap, and discontinuous. Monotonicity models consider only monotone or non-monotone, in other words swap, and discontinous are lumped together.
2. f vs. fe. The model may be conditioned on the Indonesian phrase (f), or on both the Indonesia phrase and English phrase (fe).
3. Unidirectional vs. Bidirectional. For each phrase, the ordering of itself in respect to the previous is considered. For bidirectional models, also the ordering of the next phrase in respect to the currect phrase is modeled.

We will also perform baseline evaluations. These evaluations would consist of both objective measures on statistical model perplexity and subjective measures on human judgments of quality, as well as attempts to correlate the two. We would also produce learning curves that show how system performance changes when we vary the amount of bilingual training text.

3. Implementation of English-Indonesian SMT

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.

3.2 Data 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-ID-EN (parallel corpus)

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

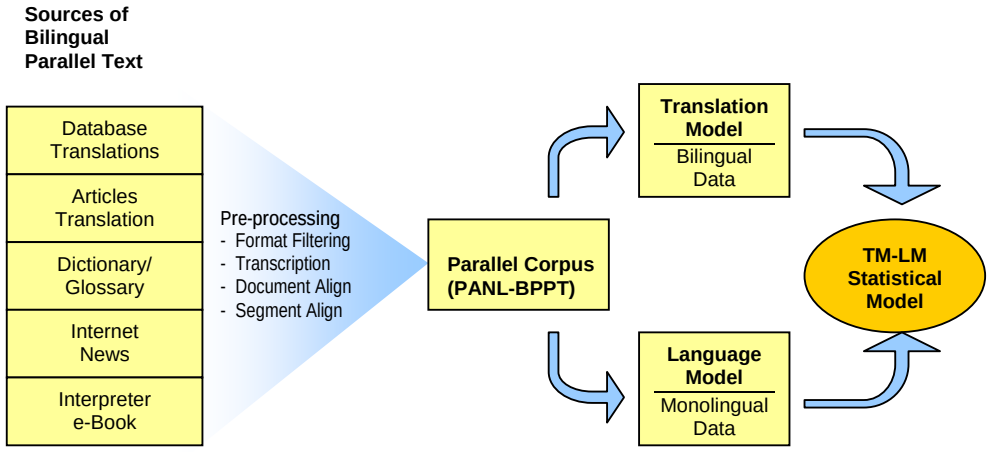

Figure 2. Data Processing of Statistical Model for SMT

Here is a brief guide on how to build the SRI LM tools and associated libraries.

1. Unpack. This should give a top-level directory with the subdirectories listed in README, as well as a few documentation files and a Makefile.
   For an overview of SRILM, see the paper in doc/paper.ps
   For reference information, look in man/html
2. Set the SRILM variable in the top-level Makefile to point to this top-level directory (an absolute path).
3. You need a SunOS (Sparc or i386), IRIX 5.x, Alpha OSF, Linux i686, Mac OSX, or CYGWIN platform to compile out of the box. For other OS/cpu combinations you will have to modify the sbin/machine-type script to detect (and name) the

platform type, and create a file common/Makefile.machine. <platform> that defines platform-dependent makefile variables.
As a workaround, the MACHINE_TYPE variable can also be set on the make comand line

```
make MACHINE_TYPE=foo ...
```

in which case no changes to sbin/machine-type are needed.
Some platform-specific notes may be found in doc/README.<platform>.
Even on the known platforms you might have to modify variables defined in common/Makefile.machine.<platform> . Candidates for changes are

`CC, CXX:` choose compiler or compiler version. For example, you might have to specify a directory path to the compiler driver.

`DEMANGLE_FILTER:` If the "c++filt" program is not installed on your system set this variable to empty.

`TCL_INCLUDE, TCL_LIBRARY:` to whatever is needed to find the Tcl header files and library. If Tcl is not available, set `NO_TCL=X` and leave the above variables empty.

It is recommended that you record changes to platform dependent variables in common/Makefile.site.<platform> and leave Makefile.machine.<platform> unchanged. That makes it easier to upgrade SRILM to future releases (just copy common/Makefile.site.<platform> to a new installation).

4. You need the following free third-party software to build SRILM:
   - gcc version 3.4.3 or higher
     (older versions might work as well, but are no longer supported).
     SRILM is occasionally tested with other compilers, see the portability notes in the CHANGES file.
   - GNU make
   - John Ousterhout's Tcl toolkit, version 7.3 or higher
     (this is currently used only for some test programs, but is needed for the build to go through without manual intervention).
   - Additional platform-dependent prerequisites are mentioned in doc/README.<os>-<machinetype>, e.g., doc/README.windows-cygwin.

The following tools are needed at runtime only:

- GNU awk (gawk), to interpret many of the utility scripts
- gzip, to read/write compressed files
- bzip2, to read/write .bz2 compressed files (optional)
- p7zip, to read/write .7z compressed files (optional)

For Windows 9x or NT, you will need the CYGWIN UNIX compatibility environment, which includes all of the above. The MinGW andVisual C platforms will also work, but with some loss of functionality. See doc/README.windows-* for more information.

Links to these packages can be found on the SRILM download page (http://www.speech.sri.com/projects/srilm/download.html).

5. In the top-level directory, run

   `gnumake World` or

   `make World` (if the GNU version is the system default)

   This will create the directories

   ```
   bin/
   lib/
   include/
   ```

   build everything and install public commands, libraries and headers in these directories. Binaries are actually installed in subdirectories indicating the platform type. To create binaries for a platform that is not the default on your system,

   ```
   use make MACHINE_TYPE=xxx, e.g.
   make MACHINE_TYPE=i686-m64 World# 64-bit binaries for Linux
   make MACHINE_TYPE=msvc World    # MS Visual C++ on Windows
   ```

6. The result of the above should be a fair number of .h and .cc files in `include/`, libraries in `lib/$MACHINE_TYPE`, and programs in `bin/$MACHINE_TYPE`. In your shell, set the following environment variables:

   ```
   PATH            add $SRILM/bin/$MACHINE_TYPE and $SRILM/bin
   MANPATH         add $SRILM/man
   ```

7. To test the compiled tools, change into the $SRILM/test directory and run
   ```
   gnumake all
   ```
   This exercises the most important (though not all) functionality in SRILM and compares the results to reference outputs. If discrepancies are reported, examine the output files in $SRILM/test/output and compare them to the corresponding files in $SRILM/test/reference.
8. After a successful build, clean up the source directories of object and binary files that are no longer needed:
   ```
   gnumake cleanest
   ```
9. (Optional) To build versions of the libraries and executables that are optimized for space rather than speed, run
   ```
   gnumake World OPTION=_c
   gnumake cleanest OPTION=_c
   ```
   The libraries will appear in ${SRILM}/lib/${MACHINE_TYPE}_c, with executables in ${SRILM}/bin/${MACHINE_TYPE}_c . The data structures used in these versions use sorted arrays rather than hash tables, which wastes less memory, but is also somewhat slower. The directory suffix "_c" stands for "compact".
   Other versions of the binaries can be built in a similar manner.
   The compile options currently supported are
   ```
   OPTION=_c       "compact" data structures
   OPTION=_s       "short" count representation
   OPTION=_l       "long long" count representation
   OPTION=_g       debuggable, non-optimized code
   OPTION=_p       profiling executables
   ```
10. Recent versions of gawk may not perform correct floating-point arithmetic unless either
    ```
    LC_NUMERIC=C    or
    LC_ALL=C
    ```
    is set in the environment. This affects many of the scripts in utils/.
11. Be sure to let me know if I left something out.

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

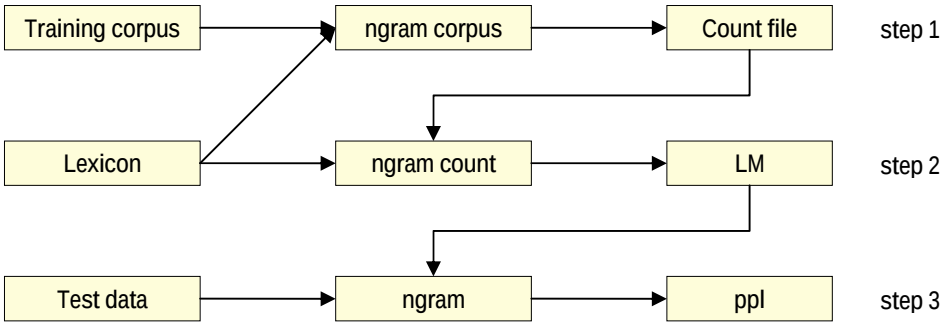

Figure 2. Step-by Step Usage of SRILM

In order to produce the Language Model (LM), we use the following command line instructions:

```
% ngram-count -text CORPUS_FILE -lm SRILM_FILE
```

The following is the sample of n-gram count which is obtained from 50.000 sentences of monolingual corpus of English:

```
\data\
ngram 1=39993
ngram 2=60900
ngram 3=40952

\1-grams:
-1.362939   </s>
-99    <s>  -0.5451546
-5.881724   aaaf
-4.896565   aachen      -0.06532428
-5.881724   aaj
-5.881724   aala
-5.881724   aamel
-5.881724   aamir
-5.881724   aan
-5.414177   aap
```

```
-5.235515   aarhus      -0.1352619
-5.001301   aaron -0.1859574
-5.881724   aarp
-5.881724   aatef
-5.881724   aater
-4.787421   ababa -0.04195464
-5.881724   abac
-5.881724   abad
-5.881724   abakr
-4.511215   abandon     -0.3002984
-5.881724   abandonall
-4.524179   abandoned   -0.1512827
-5.329418   abandoning  -0.308226
-5.626085   abandonment
-5.881724   abarge
-5.881724   abassan
-5.881724   abated
-5.881724   abating
-5.881724   abattoir
-5.881724   abayas
-5.881724   abba
-5.881724   abbar
-3.696341   abbas -0.2815464
-5.881724   abbasand
-5.626085   abbassan
-5.881724   abbey
-5.626085   abbiati
-5.881724   abbreviated
-4.741663   abc   -0.07178178
-5.626085   abd
-5.414177   abdallah
-5.881724   abdalmahmood
-4.595536   abdel -0.3050831
-5.881724   abdelaziz
-5.881724   abdelhak
-5.881724   abdelkader
.
.
.
\end\
```

3.4 Translation Model Preparation

In the following, we describe in brief on how we prepare Translation Model (English-Indonesia). There is 2 sub directories, as follows:

- Directory `bin`, which contain GIZA++ (packages which implement the IBM model) and `mkcls` which categorize the words into probability classes.
- Directory `corpus`, which corpus data is placed for training and building translation model.

In these directories, there are Perl scripts available to extract phrase table. In order to

build the phrase translation table, we use additional scripts (as attached in our report on Translation, Alignment and Other Issues Related to Corpus Development). The following command is use to build the translation model:

```
train-phrase-model.perl –root-dir . –f eng –e ind
 --corpus corpus/data20000
```

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 Pharaoh 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).

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.

In statistical MT, a key component is the translation model. We can do the same thing. An instantiated translation model, such as learned by GIZA, assigns a probability to every pair of sentences. In this case, our previously unseen test data consists of sentence pairs. Given a certain sentence e, a model will lay bets on various sentences. When the actual translation is revealed in the test data, we can see whether the model bet a lot or a little. We hope it bet a lot.

5. Future Work

The experts in translation have two differing approaches toward the translation concept: universalism and monadic. We understood there is a possibility of "un-translation" which is "translation fails – or un-translability occurs when it is impossible to build functionally relevant features of the situation into contextual meaning of target language (TL) text. Broadly speaking, the cases where this happens fail into two categories. Those where the difficulty is linguistic, and those where it is cultural.

We examine further the translability concept by taking into account that most Asian language share very similar "culture" but different in language structure. We can not enforce the system and structure to target language without "knowing" the language itself. In this case, a rule-based system should be used as a preprocessing to enable the structure of source language to approximate the structure of target language. For example, in translating Indonesian-English, we need a rule-based system to transform the DM-MD rule (Indonesian grammar rule for adjective phrase construction). This rule approximates the order of noun and adjective phrase of Indonesian according to English noun or adjective phrase. For example:

```
    MD                  DM
sebuah rumah  besar -> a big house
 (a)     (house)  (big)
gunung   biru  itu -> the blue mountain
(mountain) (blue)  (the)
```

In our future work, by implementing several modules as pre and post processors, it is expected that statistical MT will perform better in translating English to Indonesian by having a similar language structure. We described some of these approaches in the following

In a research reported in (Al-Onaizan et al., 2000), an experiment was conducted on Tetun- English translation with a small parallel corpus. The translation experiment has

been done by different groups including one using statistical machine translation. They found that the human mind is very capable of deriving dependencies such as morphology, cognates, proper names, etc. and that this capability is the crucial reason for the better results produced by humans compared to corpus based machine translation. If a program sees a particular word or phrase one thousand times during the training, it is more likely to learn a correct translation than if it sees it ten times, or never. Because of this, statistical translation techniques are less likely to work well when only a small amount of data is given. (Callison-Burch and Osborne, 2003) propose a co-training method for statistical machine translation using the multilingual European Parliament corpus. Multiple translation models trained on different language pairs are used for producing new sentence pairs. They are then added to the original corpus and all translation models are retrained. The best improvements have been achieved after two or three training rounds.

In (Nießen and Ney, 2004) the impact of the training corpus size for statistical machine translation from German into English is investigated, and the use of a conventional dictionary and morpho-syntactic information for improving the performance is proposed. They use several types of word reordering as well as a hierarchical lexicon based on the POS tags and base forms of the German language. They report results on the full corpus of about sixty thousand sentences, on the very small part of the corpus containing five thousand sentences and on the conventional dictionary only. Morpho-syntactic information yields significant improvements in all cases and an acceptable translation quality is also obtained with the very small corpus as well as using word reordering information based on POS tags.

Reference

1. Jürgen Handke, The Structure of the Lexicon: Human Versus Machine, Walter de Gruyter, 1995
2. Sonja Nießen and Hermann Ney. 2004. Statistical machine translation with scarce resources using morpho-syntactic information. Computational Linguistics, 30(2):181–204, June.
3. Brown et al, 1993 “The Mathematics of Statisti-cal Machine Translation: Parameter Estimation”, P. Brown, S. Della Pietra, V. Della Pietra, R. Mer-cer. Computational Linguistics, 19(2).
4. Chris Callison-Burch and Miles Osborne. 2003. Cotraining for statistical machine translation. In Proc. Of the 6th Annual CLUK Research Colloquium, Edinburgh, UK, January.
5. Knight, 1997 “Automating Knowledge Acquisi-tion for Machine Translation”, K. Knight, AI Magazine, 18(4).
6. Y. Al-Onaizan, U. Germann, U. Hermjakob, K. Knight, P. Koehn, D. Marcu, K. Yamada, “Translating with Scarce Resources”, AAAI-2000.
7. B. Pang, K. Knight, and D. Marcu. “Syntax-based Alignment of Multiple Translations: Extracting Paraphrases and Generating New Sentences,” NAACL-HLT-2003.

8. 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.
9. Wikipedia SMT, http://en.wikipedia.org/wiki/, retrieved August 08
10. Aston, G. and Burnard, L. The BNC Handbook Edinburgh: Edinburgh University Press., 1998
11. 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
12. Garside, R., Leech, G., and McEnery, T. Corpus annotation: linguistic information from computer text corpora, Harlow: Addison-Wesley Longman, 1997.
13. 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
14. 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).
15. 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
16. W. J. Hutchins and H. Somers. (1992). An Introduction to Machine Translation, 18.3:322. ISBN 0-12-36280-X
17. 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.
18. 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.