International Journals of Advanced Research in Computer Science and Software Engineering
ISSN: 2277-128X (Volume-7, Issue-7)

Research Article | **July 2017**

An Overview of Statistical Machine Translation Tools

Mir Aadil, M. Asger
Department of Computer Sciences, BGSBU, Jammu and Kashmir,
India

DOI: 10.23956/ijarcsse/V7I7/0201

Abstract— The process Machine translation is a combination of many complex sub-processes and the quality of results of each sub-process executed in a well defined sequence determine the overall accuracy of the translation. Statistical Machine Translation approach considers each sentence in target language as a possible translation of any source language sentence. The possibility is calculated by probability and as obvious, sentence with highest probability is treated as the best translation. SMT is the most favoured approach not only because of its good results for corpus rich language pairs, but also for the tools that SMT approach has been enhanced with in past two and half decades. The paper gives a brief introduction to SMT: its steps and different tools available for each step.

Keywords— Machine Translation, Statistical Machine Translation, SMT tools, MOSES, GIZA, SRILM, BLUE.

I. INTRODUCTION TO SMT

The pioneer of Statistical Machine Translation is Warren Weaver (1949); however the approach got through many improvements during late 1980s and early 1990s and is dominantly used these days among other approaches [1]. SMT uses huge bilingual text corpus to calculate probability of each target language sentence to be a candidate for the best translation of a particular source language sentence. Each sentence in the source language has mostly multiple translations in the target language. Depending on the context and ontology each of these target sentences gains some probability of being correct. But only one of these can be actually the real translation of the source sentence. Statistical approach selects the candidate sentence that has highest probability as the best translation.

A standard approach for estimating the conditional probability P(*t* | *s*) of a target sentence *t*, as a translation of a source language sentence *s*, is using Bayes' theorem. Using this theorem P(*t* | *s*) is rewritten as

$$\mathrm{P}(t \mid s) = \frac{P(s \mid t)P(t)}{P(s)}$$

Since *s* is constant, so omitting the denominator, P(s|t)P(t) needs to be evaluated.

So the problem of machine translation can be broken into following steps:

- Estimation of P(*t*) using a *language model*;
- Using a *translation model* for the estimation of P(*s*|*t*);
- And finally when P(*t*) and P(*s*|*t*) is estimated, using *a decoder* that can find out by searching algorithms, the target sentence *t* that results in the highest value of (P(*s*|*t*)*P(*t*)) as shown in Fig. 1.
- Finally evaluation of the translation (target sentence *t*) for accuracy.

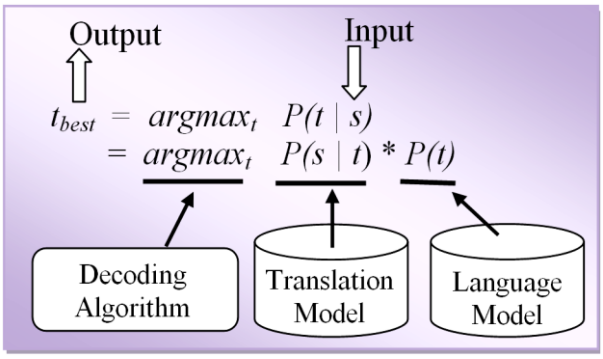

Fig. 1 Statistical Machine Translation

II. OVERVIEW OF SMT TOOLS AVAILABLE

The tools for Statistical Machine Translation are classified as per their functionality into:

A. Language Modelling Tools

The overall goal of a language model in simple terms is to find the probability of a phrase to be correct in terms of word order. A candidate sentence *t* can be split into different parts say $t = t_1 t_2 \ldots t_m$. So the probability of *t* shall be the product of the conditional probabilities of the parts, thus:

$$P(t) = \prod_{j=1}^{m} P\left(t_j \mid t_1, \ldots, t_{j-1}\right)$$

However calculating all conditional probabilities by the language model is a huge overhead. So it is assumed that presence of a word is not dependent on what was preceding it. That is assuming that

$$P\left(t_j \mid t_1, \ldots, t_{j\ 1}\right) = P\left(t_j\right), \text{ then}$$

$$P(t) = \prod_{j=1}^{m} P\left(t_j\right).$$

The probabilities $P\left(t_j\right)$need to be calculated for each $t = t_1, t_2, \ldots . t_j$ can be estimated from a really huge text corpus, and checking the frequency of each word. Thus calculating the probabilities by P $\frac{C(t_1)}{N}$, Where C(t_1)represent the number of times t_1 appears in corpus, and N is the total number of words in the corpus . This model is unigram based and may be useful but is not fully practical, that is why bigram and trigram based probability is calculated using a language model [1]. An overview of some of such Language models is given as:

1) SRILM: SRILM (SRI Language Modeling Toolkit) is used to develop and apply statistical language models (LMs), that has multiple applications like in speech recognition, statistical tagging and segmentation, and machine translation. SRI Speech Technology and Research Laboratory has been continuously working on it since 1995 providing more stable and robust versions frequently. The language models are implemented with the help of provided set of C++ class libraries, training and testing Language Models is supported by a set of executable programs. SRILM also provides some diverse scripts for other related tasks. The tool runs on UNIX and Windows platforms and is freely available for non-commercial uses. It uses n-gram statistics and also is a part of Moses and Pharaoh [2],[3].

2) IRSTLM: The IRST Language Modeling (IRSTLM) Toolkit is used to cope up with high scalability as its algorithms and data structures are suitable for large n-gram language models. IRSTLM is licensed under the GNU Library or Lesser General Public License version 2.0 (LGPLv2)[2].

3) YASMET: Yet Another Simple Maximum Entropy Toolkit with Feature Selection Version 1.0, written by Deepak Ravichandran & Franz Josef Och. This is a tiny toolkit of only 132 lines of code that performs training of maximum entropy models. The small code provides training of model parameters, evaluation, perplexity and error rate computation, count-based feature reduction, Gaussian priors, feature count normalization. Apart from being user friendly and proficient, the software is released under the GNU Public License [4].

4) MALLET: MALLET is a multipurpose, Java-based package that has applications in statistical natural language processing, document classification, clustering, topic modeling, information extraction, etc. MALLET includes well defined tools for document classification: routines for text to "features" conversion, algorithms like Naïve Bayes, Maximum Entropy, and Decision Trees. The toolkit is Open Source Software, and is released under the Common Public License[5].

5) CMU-SLM: The CMU-Cambridge Statistical Language Modeling toolkit is a suite of UNIX software tools for development and testing of LMs for large amounts of training data. Its Version 1 was written by Roni Rosenfeld and Version 2 by Philip Clarkson and Roni Rosenfeld at Carnegie Mellon University. Version 2 provides n-gram statistics support. It provides tools for word frequency lists and vocabularies, bigram and trigram counts, hit ratios and statistics, perplexity, Out-Of-Vocabulary (OOV) rate and a lot more [6].

6) KENLM: KENLM is faster and consumes lesser memory than SRILM and IRSTLM and it adjusts its data structures as per the RAM and users preference for time-speed trade-off. It needs some of the dependencies like a C++ compiler and POSIX system calls. Since querying supports n-grams, the Filtering and estimation in the tool are multi-threaded, so Boost is required for that as well as for testing. It is released under Permissive licence that is fully redistributable[14].

7) RandLM: The project is novel and is being carried out by Oliver Wilson, Alexandra Birch, David Talbot, Hieu Hoang and Miles Osborne. This projects deals n-gram-based language models built using randomized representations (Bloom Filters, etc) that is more space-efficient. It is available under a GNU General Public License version 2.0 (GPLv2) [14].

8) SALM: It is a toolkit that uses Suffix Array indexing for empirical natural language processing. It provides limited functions such as n-grams count a text corpus; however the use of a suffix array language model helps to keep a long history. It is provided with an Academic Free License (AFL) [14].

B. Translation Modelling Tools

Translation model can be created with following parameters:

- The *fertility probability* $P\left(n \mid t_i\right)$, the probability that the target word t_i has fertility n.
- The *translation probability* $P\left(s_i, a \mid t_i\right)$, one for each Source word s_i and Target word t_i with alignment a.

Where *Fertility* is the number of Target words mapped to a Source word. The value may be zero or more. *Distortion* on the other hand is the distance of a word in the source sentence and its mapping in the target language. A translation

model is defined as the probability $P(s,a|t)$. It is that for a Source sentence *s* we have a correct translation in Target i.e. a sentence *t*, with some required word alignment *a* [13],[14]. Some of the tools available for the task are:

1) GIZA++: The lot of extra features were added to GIZA tool by Franz Josef Och that was then given the name GIZA++. GIZA is a part of toolkit EGYPT. GIZA++ included support for IBM Model 4 and 5 as well as the word class dependent models for alignment. GIZA++ has the code for HMM alignment model, variants of Model 3 and 4, smoothening fertility and alignment parameters. It provides much more efficient training and pegging. It is released under the GNU Public License (GPL) [2].

2) CARMEL: It is a finite-state transducer that was implemented in C++ at Information Sciences Institute, University of Southern California by Jonathan Graehl. It serves as the translation model developing tool for PHARAOH. It has functional dependency on GNU Make, 3.8 or latter for Linux or MS Visual C++ and .NET for Windows. It supports both Bayesian (Gibbs Sampling) as well as EM (forward-backward) training along other functionalities. The product is available for academics under licence agreement with University of Southern California [2].

C. Decoders

For translation the point is how to find the Target sentence *t* which has a maximum *P(s|t)P(t)*. Having a set of candidate Target sentence $\{t_1,t_2,...t_n\}$ and a set of alignments $\{a_1,a_2,... a_m\}$,then we just need to find out a pair *(t,a)* which yields the highest value of

$$P(t,a \mid s) = \frac{P(s,a \mid t)P(t)}{P(s)}$$

Because *s* is fixed, *P(s)* in the denominator can be omitted, so the

$$P(t,a \mid s) = P(s,a \mid t)P(t)$$

The problem of translation for our study is simply to search for the pair *(t,a)* that has maximum probability *P(t,a|s)*. However, it is still not possible to carry out an exhaustive search for every sentence to be translated there are a large number of possible *t's* and the possible number of alignments for each *t* makes the condition more worse. So usually some heuristics are used e.g. a greedy algorithm that evaluates partial translations and alignments and such heuristics do aid well. Some of the popular decoders used are:

1) cdec: cdec was developed by Chris Dyer at the Language Technologies Institute in Carnegie Mellon University is a decoder, aligner, and learning framework for SMT. The Translation and alignment model is based on Finite-state transducers, SCFG, extended tree-to-string transducers(xRs). It is fast and is implemented with modular architecture in C++ with a flexible Python interface.[9]

2) ReWrite Decoder: Developed in Information Sciences Institute, University of Southern California by Daniel Marcu and Ulrich Germann, ReWrite decoder first used IBM Model 3. However, it now only supports Model 4. It used GIZA for the training purposes but since IBM Model 4 is supported by GIZA++, therefore the training is now done using GIZA++ only. Also the inputs are to be provided in XML format, as plain text isn't accepted anymore. The decoder is comparatively faster and can be customized to provide desired translations by swapping phrase boundaries for particular phrases[2].

3) MARIE: As a part of the Ph.D Thesis, Josep M. Crego developed a N-gram based SMT Decoder at the TALP Research Center of the Universitat Politecnicia de Catalunya(UPC). The decoder was designed particularly for bilingual translational units and the Translation Model trained as any typical N-gram LM as well as a typical Phrase Based Decoder as per the requirement[7].

4) Joshua: This open source decoder was developed by Chris Callison Burch in 2009 but now is officially a Apache Incubator project. It is a java written statistical machine translation decoder that can be used for phrase-based, hierarchical, and syntax-based machine translation. The decoder uses synchronous context free grammar based algorithms [2].

5) RAMSES: RAMSES is a part of MOOD translation toolkit. It is just a like PHAROAH, as it is a phrase based decoder and produces identical results to PHAROAH. RAMSES was developed mainly with the purpose to provide a decoder just like PHAROAH, freely available to researchers for use[11].

6) Phramer: Phramer is a Statistical Phrase-Based Machine Translation decoder written in JAVA with a modular architecture. It runs in Linux, Mac and Windows. The package includes code for translation table filtering, character encoding conversion, parallel decoding, sorting and profiling tools. It last version was released on 5th July, 2009 under Open Source program (BSD license)[2].

D. Evaluation Tools:

The evaluation of the resultant translation is not a very complex but important process. Manual evaluation shall look for correctness of the translation—a very harsh metric to be used. So, less hash criteria are used like fluency and adequacy. Fluency is the smoothness of the translated text in terms of grammar and idiomatic choice of words. Adequacy means the degree up to which the meaning is conveyed. It includes the evaluation of the part of message lost, added or distorted. Different tools are available for evaluation that use different parameters to rate a translation like precision, recall, f-measure, position-independent error rate, word error rate given by Levenshtein distance, etc. The most commonly tools used for evaluation are:

1) BLEU: The most commonly used automatic evaluation metrics these days is Bi-Lingual Evaluation Understudy (BLUE). It is an algorithm that works similar to position-independent word error rate, however is used to match larger n-grams with a manual reference translation. It uses precision in a modified form by calculating the ratio of the number of words of a particular translation output that match the words in a reference translation sentence to the total number of words in that output. BLUE can be also used for multiple reference evaluations, as different human beings can translate the same sentence in different ways.

2) NIST: US National Institute of Standards and Technology organizes a series of challenging research events for finding innovative machine translation evaluation metrics. Such events are known as MetricsMaTr. NIST as an evaluation tool was a result of same events. NIST use single as well as multiple references and evaluate just like BLUE. The only difference that it adjusts the weight of each n-gram based on the probability of its occurrence [8],[10],[12].

3) METEOR: Metric for Evaluation of Translation with Explicit Ordering (METEOR) uses the harmonic mean of unigram precision and recall. It is completely different approach with recall weighted higher than precision. Some unique features like stemming and synonymy matching, along with the standard exact word matching makes it an improvement BLEU metric as these features give better correlation with human judgement at the sentence or segment level and not at corpus level like BLEU.

4) ROUGE: Recall-Oriented Understudy for Gisting Evaluation (ROUGE) can be used for evaluation of machine translation. The metrics compare an automatically produced translation against a reference or a set of references just like BLEU, however, it has five diverse variants—ROUGE-N for N-gram, ROUGE-L for Longest Common Subsequence(LCS), ROUGE-W for Weighted LCS, ROUGE-S for Skip-bigrams and ROUGE-SU for Skip-bigram and unigrams

5) LEPOR: LEPOR overcomes the language bias problem as it has adjustable parameters Furthermore, in the improved version of LEPOR known as nLEPOR has n-gram support too. LEPOR is designed with the factors of enhanced length penalty for mismatches in length of translation output and reference sentence, precision to reflect accuracy, n-gram word order penalty for word order differences between the matched pair and recall for loyalty .

III. CONCLUSIONS

Most of the tools are under Open Source Software License agreement or Academic Free License and are ready available with documentation and support, it is up to the researcher to select a tool for each sub-process of machine translation. Although most of the tools are built in C++ and are tested for Linux and Unix, still compatibility of tools with each-other should be considered while selection. The space-time tradeoff as well as the compatibility of the machine on which the tools are to be used can determine the efficiency of the overall translation system. It can be concluded that efficient tools are available for Language Modeling, Translation Modeling, Decoding as well as Evaluation processes, however the final selection determining factor shall be the language pair for which the translation system is to be developed, as same tool can show different degrees of efficiency for different language pairs.

REFERENCES

[1] P.F. Brown, S. Della Pietra, "The Mathematics of Statistical Machine Translation: Parameter Estimation," *Association of Computational Linguistics*, vol.19, p. 263–311, *1993*.

[2] A. Kumar, and V. Goyal, "Comparative Analysis of Tools Available for Developing Statistical Approach Based Machine Translation System," in *Information Systems for Indian Languages. Communications in Computer and Information Science, Springer*, vol 139, 2011, pp. 254-260.

[3] A. Stolcke (2002). "Srilm —An Extensible Language Modeling Toolkit," Speech Technology and Research Laboratory SRI International, Menlo Park, CA, U.S.A. [Online]. Available: http://www-speech.sri.com/papers/icslp2002-srilm.ps.gz

[4] F. Josef. (2001) Readme-File Of Yasmet 1.0 Yet Another Small MaxEnt Toolkit: YASMET. [Online]. Available: http://www.fjoch.com/YASMET.html

[5] K.A. McCallum. (2002) MALLET: A Machine Learning for Language Toolkit. [Online]. Available: http://mallet.cs.umass.edu

[6] P.R. Clarkson, R. Rosenfeld, "Statistical Language Modeling Using the CMU-Cambridge Toolkit," in *Proceedings ESCA Eurospeech*, 1997, p. 2707–2710

[7] J.M. Crego, J. B. Mariño, A. de Gispert, "An Ngram-based Statistical Machine Translation Decoder," in *9th European Conference on Speech Communication and Technology, Lisbon, Portugal*, 2005, p. 3185–3188

[8] K. Papineni, S. Roukos, T. Ward, W.J. Zhu, "BLEU: a method for automatic evaluation of machine translation," in *40th Annual meeting of the Association for Computational Linguistics, 2002*, p. 311–318

[9] C. Dyer, A. Lopez, J. Ganitkevitch, J. Weese, F. Ture, P. Blunsom, H. Setiawan, V. Eidelman, and P. Resnik. "cdec: A Decoder, Alignment, and Learning Framework for Finite-State and Context-Free Translation Models," *In Proceedings of ACL*, July, 2010.

[10] Callison-Burch, C. Osborne, P. Koehn, "Re-evaluating the Role of BLEU in Machine Translation Research," in *11th Conference of the European Chapter of the Association for Computational Linguistics*, 2006, p. 249–256

[11] A. Patry, F. Gotti, and P. Langlais, "Mood at work: Ramses versus Pharaoh," in *Proceedings of the Workshop on Statistical Machine Translation, New York City, Association for Computational Linguistics*, 2006, p. 126–129.

[12] G. Doddington, "Automatic evaluation of machine translation quality using n-gram co-occurrence statistics," in *Proceedings of the Human Language Technology Conference (HLT), San Diego, CA*, 2002, p. 128–132

[13] K. Knight. (1990) A Statistical MT Tutorial Workbook, JHU summer workshop [Online]. Available: http://cseweb.ucsd.edu/~dkauchak/mt-tutorial/260

[14] A. Lopez. (2008) Statistical machine translation. ACM Computer Survey (2008), http://doi.acm.org/10.1145/1380584.1380586

