# Text Segmentation Criteria for Statistical Machine Translation

Mauro Cettolo and Marcello Federico

ITC-irst, Istituto per la Ricerca Scientifica e Tecnologica
via Sommarive, 18
38050 Povo di Trento, Italy
{cettolo,federico}@itc.it
http://hermes.itc.it

**Abstract.** For several reasons machine translation systems are today unsuited to process long texts in one shot. In particular, in statistical machine translation, heuristic search algorithms are employed whose level of approximation depends on the length of the input. Moreover, processing time can be a bottleneck with long sentences, whereas multiple text chunks can be quickly processed in parallel. Hence, in real working conditions the problem arises of how to optimally split the input text. In this work, we investigate several text segmentation criteria and verify their impact on translation performance by means of a statistical phrase-based translation system. Experiments are reported on a popular as well as difficult task, namely the translation of news agencies from Chinese-English as proposed by the NIST MT evaluation workshops. Results reveal that best performance can be achieved by taking into account both linguistic and input length constraints.

## 1 Introduction

Current machine translation (MT) systems are in general unable to process long texts in a single step. Long documents are typically split into smaller and more manageable chunks, here simply called segments. For the sake of our exposition, a segment is here a generic sequence of words, not necessarily corresponding to a linguistic unit.

At first sight, text segmentation based on linguistic criteria should be the best choice; however, any segmentation method should also take into account the way a specific system works.

Statistical MT (SMT), for instance, typically relies on a beam search algorithm to control the growth of the solution space, and on statistical models or feature functions to compute scores of translation hypotheses. Moreover, several SMT systems use multi-stage decoding strategies. That is, the search algorithm first generates a list of N-best translation candidates, then these translations are re-scored and re-ranked by means of additional and richer feature functions. In this framework, the length of the input string plays indeed a relevant role. The longer the input string, the more drastic will be the cut of hypotheses by

the beam. Moreover, at a fixed length N of the N-best list, the longer the input string the less the list will represent the solution space. From the point of view of efficiency, shorter segments can in generally lead to faster and less memory consuming translation and better exploitation of multi-processing resources.

The above issues would suggest to opt for short input strings; however, statistical models applied in SMT can deliver better translation quality if sufficient context is available for all words in the input. Hence, requirements by the statistical models act in opposition to those of the search algorithm.

In this work, we investigate different text segmentation criteria and look at their impact on translation performance by using a state-of-the-art phrase-based SMT system. The investigated methods address real working conditions, such as the translation of documents or spoken language, where the text to be translated is produced by a speech recognizer. In this case, linguistically motivated segments are difficult to obtain, given the difficulty to reliably detect sentence boundaries from linguistic and acoustic cues.

The basic goal of this work is to understand if better performance can be gained by combining linguistically motivated criteria with length-based methods that take into account the peculiarities of the used SMT system.

The paper is organized as follows. In Section 2, several aspects of the statistical SMT system developed at ITC-irst are introduced, namely: the statistical model, the system architecture, some details on the decoding algorithm and rescoring module, and finally its domain. In Section 3, the segmentation types to be compared are presented and their pros and cons are commented. Finally, Section 4 shows and discusses results. A section with conclusions ends the paper.

## 2 Phrase-based Translation System

Given a string $\mathbf{f}$ in the source language, the goal of statistical machine translation is to select the string $\mathbf{e}$ in the target language which maximizes the posterior distribution $\Pr(\mathbf{e} \mid \mathbf{f})$. In phrase-based translation, words are no longer the only units of translation, but they are complemented by strings of consecutive words, the phrases. By assuming a log-linear model [1, 2] and by introducing the concept of word alignment[3], the optimal translation can be searched for with the criterion:

$$\tilde{\mathbf{e}}^* = \arg\max_{\tilde{\mathbf{e}}} \max_{\mathbf{a}} \sum_{r=1}^{R} \lambda_r h_r(\tilde{\mathbf{e}},\ f, \mathbf{a}),$$

where $\tilde{\mathbf{e}}$ represents a string of phrases in the target language, $\mathbf{a}$ an alignment from the words in $\mathbf{f}$ to the phrases in $\tilde{\mathbf{e}}$, and $h_r(\tilde{\mathbf{e}}, \mathbf{f}, \mathbf{a})$ $r = 1, \ldots, R$ are *feature functions*, designed to model different aspects of the translation process.

The assumed translation process extends step by step the target string by covering new source positions until all of them are covered. For each added target phrase, a source phrase within the source string is chosen, and the corresponding score is computed on the basis of its position and phrase-to-phrase translation probabilities. The fluency of the added target phrase with respect to its left

context is evaluated by a 4-gram language model. Some exceptions are also managed: target phrases might be added which do not translate any source word, and some of the source words can be left untranslated, that is they are translated with a special empty word.

### Model Training

The resulting log-linear model embeds feature functions whose parameters are either estimated from data or empirically fixed. The scaling factors $\lambda$ of the log-linear model are instead estimated on a development set, by applying a *minimum error training* procedure [4, 5].

The language model feature function is estimated on unsegmented monolingual texts.

The phrase-to-phrase probability feature is estimated from phrase-pair statistics extracted from word-aligned parallel texts. Alignments are computed with the GIZA++ software tool [6] which implements statistical models developed by [3, 6]. Phrase pairs are extracted from the segment pairs by means of the algorithm described in [7].

For the sake of this work, it is worth explaining how phrase-pair statistics are extracted. Since long parallel texts represent a problem in training word-alignment models, they are split into smaller parallel segments by means of a binary and recursive procedure. The method relies on a likelihood measure which evaluates the correspondence of a segment pair in the source and target language, respectively. Text break candidates are chosen both according to strong punctuation and segment length. For our training, the final length of parallel segments is at most 30 words.

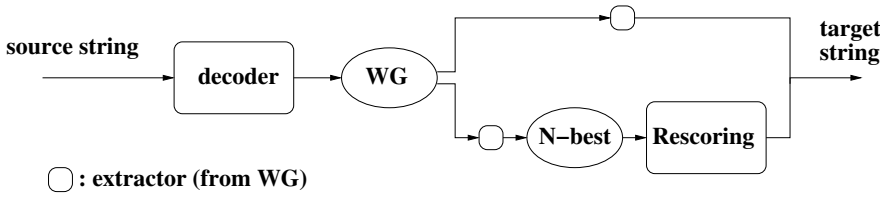


**Fig. 1.** Architecture of the ITC-irst SMT system. The decoder produces a word-graph (WG) of translation hypotheses. In single-stage translation the most probable string is output. In two-stage decoding, N-best translations are extracted, re-scored, and re-ranked by applying additional feature functions

### Decoding Strategy

Figure 1 illustrates how the translation of an input string is performed by the ITC-irst SMT system [7]. In the first stage, a beam search algorithm (decoder)

computes a word graph of translation hypotheses. Hence, either the best translation hypothesis is directly extracted from the word graph and output, or an N-best list of translations is computed by means of an exact algorithm [8]. The N-best translations are then re-ranked by applying additional features and the top ranking translation is finally output. Additional feature functions include: IBM models 1 and 3, a broad 5-gram LM and task-specific 3-gram LMs.

The decoder [9] exploits dynamic programming, i.e. the optimal solution is computed by expanding and recombining previously computed partial theories. Theory expansion basically follows the translation process explained above.

To cope with the large number of generated theories, a beam is used to prune out partial theories that are less promising and constraints are set to possible word re-ordering.

Pruning is applied on all theories covering the same set of source positions, and on all theories with the same output length.

Word re-ordering constraints are applied during translation each time a new source position is covered, by limiting the number of vacant positions on the left and the distance from the left most vacant position. In the following experiments, both parameters were set to 3, which results in a good compromise between quality and speed.

**Table 1.** Statistics (number of words) of training and test data. The size of the English side of the test set refers only to the gold reference

| | parallel resources | | monolingual resources |
|---|---|---|---|
| | Chinese | English | English |
| training | 82M | 88M | 464M |
| test | 26K | 29K | – |

### Translation Task

The task considered in this paper is the translation of news agency texts from Chinese to English as proposed in the NIST MT Evaluation Workshops.[1] The ITC-irst system was trained according to the so-called large data condition. Table 1 gives figures about training and test corpora, which also include punctuation marks in both languages. As testing data we used the NIST 2003 evaluation set.

Translation performance are reported in terms of BLEU [10] and NIST [11] scores, that were computed with the case-insensitive modality and by exploiting four reference translations.

Since segment boundaries of the reference translations are those prepared by the Linguistic Data Consortium (LDC), the problem arises of how to score

[1] `www.nist.gov/speech/tests/mt/.`

translations computed with different segmentations of the input. A reasonable solution is provided by a publicly available tool developed by RWTH [12], which automatically aligns, with possible errors, the translation hypotheses to the multiple reference translations.

## 3 Segmentation Criteria

The segmentation criteria we investigated are the following.

**Linguistic-based Criteria**

- *ldc*: original segmentation provided by LDC. Most segments end with a strong punctuation mark, but not all of them. Possibly, sentences separated by strong punctuation marks are joined into a single segment: this happens when sentences are semantically tied.
- *strongPnct*: segmentation obtained by splitting the input text stream on strong punctuation (“.”, “!” and “?”). Possibly, segments can be either very short or very long. Segments do not contain semantic breaks, but it happens that contiguous sentences are split even if they are semantically related.

**Length-based Segmentation**

- *fixedLEN*: segmentation obtained by cutting the text into segments of fixed length LEN, whatever are the tokens around each break. Breaks are not linguistically motivated, moreover feature functions of the SMT system are not expected to model well words on the segment boundaries. On the other side, since decoder computes hypotheses of about the same length for each input segment, all N-best lists should have similar content variability, allowing an easier qualitative evaluation of the re-ranking module.

**Combined Criteria**

- *pnct&lenLEN*: this is a linguistically refined version of the *fixedLEN* segmentation. It is obtained by first looking for strong punctuation and then for weak punctuation (“,”, “;”, “:” and “-”) within a window (of size LEN) centered at distance LEN from the beginning of the segment. If no punctuation mark is found, the segment size is set to LEN. Differently from *strongPnct* segmentation, segments can neither be very short nor very long (according to the LEN value). Most breaks are linguistically motivated and the average segment length can be tuned by means of the LEN value.
- *pnct&maxLEN*: segmentation obtained by further splitting segments of *strongPnct* segmentation which are longer than LEN. They are split on weak punctuation, if present, or anyway at length LEN. The rationale behind this type of segmentation is the elimination of long segments from *strongPnct* which cause long decoding time and low variability inside the N-best lists.

## 4 Results

Table 2 collects results of all experiments. Each line refers to a complete translation run of the test set segmented according to one of the segmentations described above; for those depending on the parameter LEN, the most significant LEN values have been tested. For each translation run, the following numbers are supplied:

- total number of segments and their minimum, average and maximum length;
- BLEU and NIST scores of the first best output by the decoder;
- total number of theories generated during decoding; this value is approximately a linear function of the decoding time, but it is preferable to that as it is independent from the hardware;
- BLEU and NIST values of the highest scored translation hypothesis after the re-ranking of the 5000-best lists provided by the decoder;
- performance gain due to the re-scoring stage.

In the following subsections, results are commented.

**Table 2.** Performance measured with different types of source text segmentation

| SegType | #Seg | SegLength | | | Decoder | GenTh | Rescoring | Rescoring $\Delta$ |
|---|---|---|---|---|---|---|---|---|
| | | min | avg | max | bleu/nist | ($\times 10^9$) | bleu/nist | bleu/nist |
| **linguistic-based criteria** | | | | | | | | |
| ldc | 919 | 4 | 27.8 | 93 | 29.22/8.841 | 1.23 | 30.96/9.060 | +1.74/+.219 |
| strongPnct | 825 | 3 | 31.0 | 103 | 28.79/8.764 | 1.25 | 30.52/9.006 | +1.73/+.242 |
| **length-based segmentation** | | | | | | | | |
| fixed10 | 2558 | 10 | 10.0 | 11 | 24.36/8.207 | 0.39 | 24.85/7.979 | +0.49/ –.228 |
| fixed20 | 1279 | 20 | 20.0 | 21 | 26.55/8.498 | 0.85 | 28.33/8.609 | +1.78/+.111 |
| fixed31 | 825 | 31 | 31.0 | 32 | 27.30/8.588 | 1.24 | 28.95/8.782 | +1.65/+.194 |
| fixed40 | 639 | 40 | 40.0 | 41 | 27.80/8.658 | 1.36 | 29.10/8.833 | +1.30/+.175 |
| fixed50 | 511 | 50 | 50.0 | 51 | 28.05/8.705 | 1.41 | 29.29/8.888 | +1.24/+.183 |
| fixed60 | 426 | 60 | 60.0 | 61 | 27.80/8.666 | 1.44 | 29.01/8.867 | +1.21/+.201 |
| fixed70 | 365 | 70 | 70.0 | 71 | 28.08/8.690 | 1.47 | 29.43/8.889 | +1.35/+.199 |
| **combined criteria** | | | | | | | | |
| pnct&len20 | 1265 | 11 | 20.2 | 29 | 28.31/8.720 | 0.91 | 30.00/8.850 | +1.69/+.130 |
| pnct&len30 | 840 | 16 | 30.5 | 44 | 28.73/8.777 | 1.23 | 30.75/9.046 | +2.02/+.269 |
| pnct&len50 | 510 | 27 | 50.2 | 74 | 29.04/8.807 | 1.43 | 30.45/9.023 | +1.41/+.216 |
| pnct&len70 | 367 | 38 | 69.7 | 103 | 28.90/8.794 | 1.49 | 29.89/8.985 | +0.99/+.191 |
| pnct&max40 | 1082 | 2 | 23.6 | 41 | 28.72/8.793 | 1.11 | 30.56/9.011 | +1.84/+.218 |
| pnct&max50 | 948 | 2 | 27.0 | 51 | 28.89/8.793 | 1.18 | 30.74/9.031 | +1.85/+.238 |
| pnct&max60 | 875 | 2 | 29.2 | 61 | 28.98/8.809 | 1.23 | 30.63/9.029 | +1.65/+.220 |

### 4.1 Linguistic-based segmentation

The *ldc* segmentation has segments whose length is very variable (4 to 93 words) and about 28 on average. Performance of both decoder and re-scoring stages are good, yielding to the best global BLEU and NIST scores. Probably, this is because breaks were selected by humans on a linguistic basis; anyway, one must also consider that the system was trained and tuned on data provided by LDC, which processed in a coherent way also the evaluation set.

The quality of the decoder output generated on strong punctuation (*strongPnct*) is lower than that generated from *ldc*. This reveals the importance of keeping in the same segment sentences which are semantically related even if they are separated by a strong punctuation mark. Length of segments of *strongPnct* is quite similar to that of *ldc*; this is why the gains of the second stage are comparable.

### 4.2 Length-based segmentation

Fixed length segmentations were tested for lengths ranging from 10 to 70 words. As expected, the trend shows that the longer the segments, the higher are the translation quality and computational cost by the decoder.

The decoder improves its performance up to 50-word segments; long segments mean few segment boundaries, that is little processing with poor translation context. With segments longer than 50 words, a performance saturation is observed. Anyway, the quality of the decoder output is definitely worse than the translations generated from linguistic-based segmentations. Hence, the decoder models seem to suffer from random breaks.

Concerning the re-scoring stage, the longer the segments the lower is the gain, since the variability of the 5000-best lists reduces. One exception is the low performance increment observed with very short segments (10 words). This is probably due to the large number of segment boundaries where words are difficult to translate due to the lack of context. For segments of length comparable to that of *ldc* and *strongPnct* segmentations, the re-scoring gain is similar. This tells that for the sake of re-scoring, the length of segments is at least as important as having linguistically motivated breaks.

### 4.3 Linguistic- and length-based segmentation

The *pnct&lenLEN* segmentations produce segments whose length is less variable than linguistic-based segmentations.

The decoder guarantees good quality, thanks to the non-randomness of sentence breaks. Translation quality tends to increase by increasing the length of segments, as for fixed length segmentation. Hence, for the sake of the decoder it seems to be preferable to reduce as much as possible the number of breaks.

The re-scoring module works well. In particular, when segments include on average around 30 words, BLUE and NIST scores increase, respectively, by 2.02% and 0.269, which are the highest advances observed in our experiments. This

outcome together with the good re-scoring performance for *fixed20* and *fixed30* segmentations show that the re-scoring stage improves if segments are: (i) not too short in order to have an high number of plausible translations; (ii) not too long so that N-best lists include many different translations; (iii) linguistically motivated; and (iv) coherent with training and system tuning conditions.

The behavior of *pnct&maxLEN* segmentations is clear. The decoder performs well, favored also by the match with training and tuning conditions. Re-scoring is good since breaks are linguistically motivated, but not so much as for *pnct&lenLEN* segmentations, due to the presence of very short segments (up to 2 words). The split of *strongPnct* long segments results to be useful both for decoder and re-scoring stage.

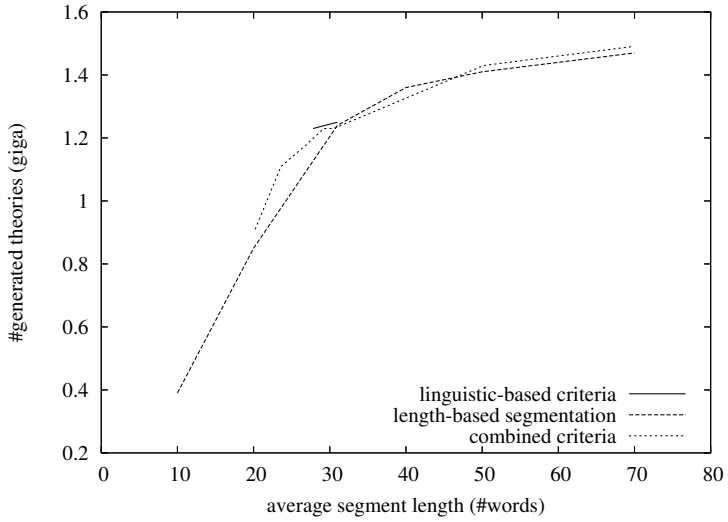


**Fig. 2.** Generated theories (decoding time) as a function of the average segment length for the segmentation criteria under investigation

### 4.4 Decoding time vs. segment length

Figure 2 plots the number of generated theories, which is proportional to decoding time, versus the average segment length for each segmentation criteria analyzed in this work. It is evident that the decoder searches portions of the search space whose size depends only on the average number of words to be translated, and not on the segmentation criterion. Moreover, the impact of the beam approximation is plain: in spite of the exponential growth of possible translations with the input length, the corresponding number of actually generated theories tends to saturate, proving that the cut of theories by the beam search is

larger for longer inputs. Hence, it could happen that for very long inputs the decoder performance degrades. However, this phenomenon was not observed with the here considered segment lengths.

## 5 Conclusions

Current MT systems are unable to process huge blocks of text in one shot. Input text stream must be split in manageable segments. Hence, the problem arises of how to automatically segment the input text. Linguistic-based criteria are expected to work well in theory, but in practice segments should also suit the features of the used MT system. In statistical MT it is known that segment length affects the behavior of the search algorithms and its embedded statistical models.

In this work, we dealt with the problem of source text segmentation with respect to a state-of-the-art phrase-based SMT system, based on a two stage decoding strategy.

The quality of translations was measured when these were originated from different input segmentation types: linguistic-based, length-based, and a combination of the two.

Results reveal that it is important to break the source text stream by fulfilling linguistic constraints, but performance of a real SMT system can be improved by providing segments of adequate length. From the decoder perspective, long segments favor translation quality. From the re-scoring point of view, the length of segment should balance content variability inside the N-best list and the matching of conditions used to train the system.

## Acknowledgments

This work has been funded by the European Union under the integrated project TC-STAR - Technology and Corpora for Speech to Speech Translation - (IST-2002-FP6-506738, http://www.tc-star.org).

## References

1. A. Berger, S. A. Della Pietra, and V. J. Della Pietra. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1):39–71, 1996.
2. F. J. Och and H. Ney. Discriminative training and maximum entropy models for stati stical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, PA, Philadelphia, USA, 2002.
3. P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–312, 1993.
4. Franz Joseph Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics (ACL)*, pages 160–167, Sapporo, Japan, 2003.

5. M. Cettolo and M. Federico. Minimum Error Training of Log-Linear Translation Models. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, pages 103–106, Kyoto, Japan, September 2004. http://www.slt.atr.jp/IWSLT2004/archives/000619.html.
6. F. J. Och and H. Ney. Improved Statistical Alignment Models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, Hong Kong, China, 2000.
7. B. Chen, R. Cattoni, N. Bertoldi, M. Cettolo, and M. Federico. The ITC-irst SMT System for IWSLT-2005. In *Proceedings of IWSLT*, 2005. http://www.is.cs.cmu.edu/iwslt2005/proceedings.html.
8. B.H. Tran, F. Seide, and V. Steinbiss. A Word Graph based N-Best Search in Continuous Speech Recognition. In *Proceedings of ICLSP*, 1996.
9. Marcello Federico and Nicola Bertoldi. A word-to-phrase statistical translation model. *ACM Transaction on Speech Language Processing*, 2(2):1–24, 2005.
10. K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a Method for Automatic Evaluation of Machine Translation. Research Report RC22176, IBM Research Division, Thomas J. Watson Research Center, 2001.
11. G. Doddington. Automatic Evaluation of Machine Translation Quality using N-gram Co-occurrence Statistics. In *Proceedings of the ARPA Workshop on Human Language Technology*, 2002.
12. E. Matusov, G. Leusch, O. Bender, and H. Ney. Evaluating Machine Translation Output with Automatic Sentence Segmentation. In *Proceedings of IWSLT*, 2005. http://www.is.cs.cmu.edu/iwslt2005/proceedings.html.