A New Quantitative Quality Measure for Machine Translation Systems

Keh-Yih Su[†], Ming-Wen Wu[‡] and Jing-Shin Chang[†]

[†]Department of Electrical Engineering
National Tsing-Hua University
Hsinchu, Taiwan 300, R.O.C.
email:kysu@ee.nthu.edu.tw, shin@ee.nthu.edu.tw

[‡]Behavior Design Corporation
2F, 28, R&D Road II
Science-based Industrial Park
Hsinchu, Taiwan 300, R.O.C.

ABSTRACT

In this paper, an objective quantitative quality measure is proposed to evaluate the performance of machine translation systems. The proposed method is to compare the raw translation output of an MT system with the final revised version for the customers, and then compute the editing efforts required to convert the raw translation to the final version. In contrast to the other proposals, the evaluation process can be done *quickly* and *automatically*. Hence, it can provide a quick response on any system change. A system designer can thus quickly find the advantages or faults of a particular performance-improving strategy and improve system performance dynamically. Application of such a measure to improve the system performance *on-line* on a parameterized and feedback-controlled system will be demonstrated. Furthermore, because the revised version is used directly as a reference, the performance measure can reflect the real quality gap between the system performance and customer expectation. A system designer can thus concentrate on practically important topics rather than on theoretically interesting issues.

1. Introduction

There are several reasons performance measure is required while building machine translation systems. (1). Potential customers need to be able to compare the performance of different systems. (2). System designers would like to keep abreast of the current system performance, and make sure the system keeps on improving, not subject to the *flip-flop* problem. (The flip-flop problem is caused when system designers try to fix some problem of the system without a thorough test. While that problem is solved, other problems may pop up. This kind of problem becomes more serious when the system scales up.) (3). The measure feedback will highlight correct research direction. (4). In a parameterized system, if a quantified cost function is provided, it can be used directly for parameter tuning. Thus, a systematic and standardized approach for performance evaluation and the establishment of a common testing base are urgently required.

Most conventional approaches evaluate system performance by human inspection and subjective measures. While post-editing, the post-editors can provide feedback on the quality of machine translation, which then is used for dictionary update and linguistic analyses of errors [Ross 82]. Also, feedback can be obtained from professional translators who annotate carefully on the printouts of raw translation output [Pigo 82]. From human feedback, system designers tend to overtune the system. Another approach, proposed in [King 90], is to collect the test suites and divide them into two sections: one to look at source language coverage, the other to examine translational problems. Such an approach can avoid the over-tuning problem caused by human feedback. The advantage of human inspection is that humans can pinpoint the real linguistic problems and make corrections. However, there are several disadvantages: (1). It is too costly for human inspection of the translation output quality. To get significant statistics on the real system performance, a large volume of text must be provided. The cost for human inspection is thus extremely high. (2). It will take too long for the results to come out. For this reason and the cost consideration, it can not be repeated frequently. Therefore, it can not provide a quick suggestion to a system designer when the system is changed or when the domain is alerted. For a system that must handle a wide variety of types of text, it fails to provide immediate help to adapt to the particular domain or field. (3). It is not easy to achieve consistency and objectiveness. Even for the same person, it is very likely that he/she would judge a translation result differently at different time, especially when the evaluation criteria are loosely defined.

Based on the above problems with human inspection, some automatic approaches were proposed to evaluate translation output quality. In [Yu 91], for example, a corpus of 3,200 sentences were collected. Then, some test points are selected by linguists based on the sentences in the corpus. The *test points* are what linguists think the most important features for the sentences in the corpus. Each test point is assigned a weight according to its importance in translation. The test points are coded in programs, therefore the testing can be done automatically. The advantage of this approach is that since their criteria are purely linguistic, they can do a very delicate evaluation and find the real linguistics problems involved. However, to acquire significant statistics on the performance, a large corpus is required. Corpus collecting and test points selecting are very time-consuming. Furthermore, to achieve high grade in quality with respect to these test points, the system might be over-tuned to the set of particular test points such that they fail to reveal their real performance on a broader domain. The system designer might thus be misled by such a close-test or training-set performance and have an over-optimistically evaluated figure of performance. (See [Devi 82, Chapter 10] for detailed comments on performance evaluation.)

We propose a new quantitative quality measure to evaluate the performance of machine translation systems. The method is to compare the raw translation output of an MT system with the final revised version for the customers, and then the editing efforts required to convert the raw translation to the final version is computed. Compared with the above proposals, the evaluation process can be done quickly and automatically. Moreover, application of such a measure to improve the system performance on-line on a parameterized and feedback-controlled system is easy. Finally, since the revised version is used directly as a reference, the performance measure can reflect the real quality gap between the system performance and customer expectation.

2. Performance Evaluation Using Bi-Text Corpus

2.1. Criteria for a Good Measure

From the above discussion, it is desirable to have a performance measure and a performance evaluation process with the following properties:

[1] low cost: minimal human interference is involved and can be done automatically.

[2] high speed: it can give system designers quick response and immediate help (even *on-line*, for a parameterized system); it can also provide positive stimulation to the system designer psychologically

[3] exactness: the difference between customer expectation and real system performance can be reflected.

Because the design goal of a system is to optimize some gain or minimize some cost, a good performance measure is definitely an important factor on the improvement of the system.

2.2. A Distance Measure Approach

To achieve the goals outlined in the previous section, a quantitative measure is proposed. In our approach, we first establish a *bi-text corpus* composed of source language sentences and the corresponding target language sentences. The target sentences are the revised version of the raw translation which were post-edited to tailor to the customers' need (for publication). Therefore, the target sentences are what customers really want. Then, we employ a *distance measure method* to evaluate the minimum distance between the raw translation output and the target sentences in the bi-text corpus. By *distance*, we mean the editing efforts needed to edit the raw translation output into the revised version. In other words, we would like to know the number of key strokes required to be performed for such editing. The smaller the distance is, the higher the translation output quality is.

The sentence pairs in the bi-text corpus is the source sentence and the target sentence post-edited for the customers. The reason for adopting the revised version text as the measure reference is that even the machine translated texts are error-free judged by system designers, it may not be the final version customers really want. In general, the system designers, who are aware of the limitation and restriction of an MT system, tend to give loose quality criteria, and thus an overoptimistic evaluation. Human inspection can only achieve correctness and readability, but the acceptability to customers is usually low. We try to offer customers the solution they really need. Thus, every trial to fine-tune the output quality should be directed to fit customers' needs [Chen 91, Wu 91].

This approach has several advantages over other methods: (1) Since the final revised version is used for comparison, it will reflect the real quality gap between the capability of the system and the expectation of the customers. According to our experience on providing translation services with the ArchTran MTS, for most translation materials, even for manuals or announcement, the final versions are intended for *publication*, not just for information retrieval. Therefore, traditional quality measures which are graded loosely like 'correct', 'understandable', ... and so on, provides little information on how the system should be tuned. Thus, it's reasonable to adopt the final revised version as the measure reference. (2) Human power is more expensive than computer power. Since this approach involves no human interference, the evaluation cost is fairly low. (3) The current system performance can be reported very soon because of high computer speed. With the quick feedback, more performance improving strategies can be tried out, and thus research efficiency is improved. (4) We can show improvement to raise research morale and excite enthusiasm, for a clear indicator of performance improvement is the strongest incentive for R&D engineers. System problems can be located and solved quickly, thus a lot of work is saved. (5) Because the final revised version is used as the measure reference, the text can be classified into different domains and styles. With the quick feedback, it can help adapt system to different domains and styles.

2.3. Distance Measure and Weight Assignment

Four primitive editing operations are defined, namely *insertion, deletion, replacement* and *swap* [Wagn 74, Lowr 75]. Since each operation requires different editing efforts, different weights must be assigned. We assign the weight according to the number of key strokes needed for each operation under the most popular editor. For Chinese editing, the weights we assign for *insertion* is 5, *deletion* 1, *replacement* 5 and *swap* 6. The deletion operation is the least costly operation for its simplicity. The insertion and replacement operations take more efforts including cursor addressing, entering and leaving editing mode. The swap operation needs a little more effort than insertion and deletion. (The swap cost is defined here to be the cost of one insertion plus one deletion. For a post-editing facility with a special swap editing function, the swap cost should be a function of the distance between the characters to be swapped. This cost might be less than the cost of one insertion plus one deletion for adjacent words. For the present, the cost is used for simplicity.)

2.4. Alignment

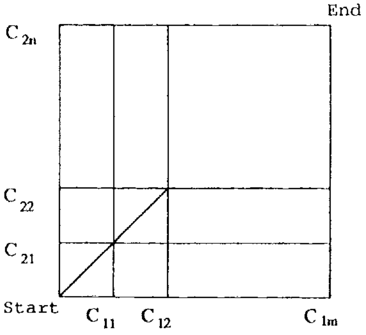

Figure 1. Alignment of raw output sentence with revised version sentence

The evaluation of the distance between the raw output sentence and the final revised sentence can be formulated as a "shortest path" searching problem. The problem can be solved with the well-known *dynamic programming technique* [Bell 57]. Figure 1 shows a diagram for the dynamic programming problem. Assume that $R = \{c_{11}, c_{12}, ..., c_{1m}\}$ is the raw output sentence composed of m characters c_{11} through c_{1m}, and $Q = \{c_{21}, c_{22}, ..., c_{2n}\}$ is the final version sentence composed of n characters c_{21} through c_{2n}.

In Figure 1, the big square has $m \times n$ grids with weight (cost or distance) associated with each edge and diagonal. Many paths can be picked from the Start to the End. Any path along the edge or diagonal from the Start to the End represents a sequence of editing operations that changes the raw output sentence to the final revised sentence. The cost incurred for each path is the accumulative weights along the path travelled. The cost/distance of a path stands for the editing efforts to make the changes. Therefore, the minimum distance path stands for the least cost. The goal is to pick the path with the minimum cost, or shortest distance.

There are three directions to go at each position: right, up or up-right. We can make an analogy between finding the shortest path and performing the fewest editing operations to convert the raw output sentence into the final version sentence. When we are at the Start point, we have the raw output sentence. If we go right, a deletion operation is performed. If we go upward, an insertion is performed. If we go along the diagonal, either one of two cases will happen. When c_{1i} and c_{2j} on the two edges of the diagonal are the same, no operation is performed, and no cost is incurred. If, however, they are different, a replacement is performed. When we eventually reach the End point, we have edited the raw output sentence into the final version sentence. The second path traversal is required to compute the number of *swap* operation. If deletion of one character is to be performed, and that character will have to be inserted in the following operations, then one deletion and one insertion are replaced by one swap operation. By the same token, If the inserted character will be deleted later, the insertion and deletion are saved by performing one swap. If the shortest path is picked, then we have edited the sentence with least effort.

The distance between the raw output sentence and the revised sentence can be formulated as follows:

$$D \equiv w_i \times n_i + w_d \times n_d + w_r \times n_r + w_s \times n_s$$

where n_i, n_d, n_r and n_s are the numbers of operation for insertion, deletion, replacement and swap performed respectively; w_i, w_d, w_r and w_s are the weights for these operations. D is the total distance for one specific editing sequence; that is to say, D is the number of key strokes required to post-edit the raw translation sentence into the final version sentence.

2.5. An Example

The solid path in figure 2 gives an example to show the steps performed using dynamic programming to find the least cost for editing one sentence. The raw output sentence is "This is my own computer" in the X axis, and the revised version sentence is "This computer is mine" in the Y axis. One insertion and deletion along the path are marked an "X" because the word "computer" appears in different locations in two sentences. Therefore, a swap operation is performed to save one insertion and one deletion. Totally, there are one replacement ("my" to "mine"), one deletion ("own") and one swap ("computer"). Table 1 shows the path with the least cost. The first row in Table 1 is the raw output sentence, the second row the revised sentence, and the third the editing operations performed. The least cost is 12 (i.e., $1 \times 5 + 1 \times 1 + 1 \times 6 = 12$). And, the average cost is 2.4 per word. Note the difference between such a measure with a conventional subjective approach. The two sentences might be judged as equally *readable* and *understandable* by human inspection. However, to tailor to the final output for publication, we still need 2.4 units of cost per word. If we follow the dotted path in Figure 2, we will get the path not of the least cost. Table 2 shows this path. In this case, there are one deletion and three replacements. The cost incurred is 16 (i.e., $1 \times 1 + 3 \times 5 = 16$).

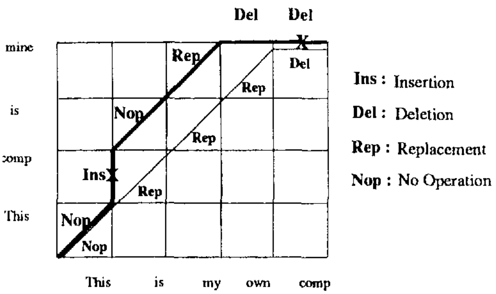

Figure 2. An example to show the steps in dynamic programming

Raw	This		is	my	own	comp
Rev	This	comp	is	mine	X	
Edit Op	NOP	SWAP	NOP	REP	DEL	

Table 1. A path of the least cost 12

Raw	This	is	my	own	comp
Rev	This	comp	is	mine	X
Edit Op	NOP	REP	REP	REP	DEL

Table 2. A path of cost 16

3. Application to Performance Evaluation and Improvement

As discussed above, the most direct application of the preference measure is, of course, to show the current status of the system performance. This function directly serves several purposes.

[1] With this performance measure applied to a large bitext corpus, one can show to the potential customers the current system performance in terms of the editing efforts required to get *high quality* translation. Furthermore, because the performance measure is an objective measure, it can be used to compare the system performance with other systems based on the same testing base.

[2] The quick response makes it possible for the system designers and the linguists to get a clear idea about the advantages or faults of a particular strategy or formalism. From the quick feedback of the measurement, one can try different approaches in rather short time. Hence, the research pace will be accelerated rapidly. And the system designers can make sure the system is on the right track.

[3] Psychologically, a clear indicator of performance improvement is the strongest incentive for R&D teams. According to our working experience, the research team members tend to become upset when their ideas can not be fully implemented and justified in a reasonable time. With a clear performance indicator and quick response, the team members usually get excited and their morale is raised substantially.

The following case study shows how the quick and automatic performance evaluation method help make decision on some designing issues and highlight research directions. In a recent evaluation run, a bi-text corpus, containing 6,110 English-Chinese sentence pairs are used to evaluate a particular version of the ArchTran English-Chinese MT system. The Chinese sentences are the revised version of the corresponding English sentences, which are to be published as a Chinese technical manual. The revised Chinese sentences are used as the reference for comparison with the unrevised version. The editing effort required to post-edit the unrevised version is then evaluated using the proposed distance measure. It takes only about *30 seconds* to get the required measure. The experimental results are shown in Table 3.

At first, we think the editing cost might be too high to get the required high quality, and we suspect that the probabilistic disambiguation mechanism for the analysis module [Chan 92a, Liu 90, Su 88, 89, 91b] might not be properly tuned. So we use an adaptive learning algorithm [Amar 67, Kata 90, Su 91a] to adjust the probabilistic disambiguation modules of the system. Table 4 shows the comparison of the original status of the MTS and its best-tuned case. In the best case, the translation with the least cost is selected.

total number of sentences	6,110
total number of Chinese characters	135,318
total number of insertions	23,541
total number of deletions	21,238
total number of replacements	22,721
total number of swaps	9,953
number of insertions per sentence	3.85
number of deletions per sentence	3.48
number of replacements per sentence	3.72
number of swaps per sentence	1.63
total cost for 6,110 sentences	312,266
cost per sentence	51.11
cost per character	2.31

Table 3. Statistics summary of distance measure (Editing cost scheme: Insertion 5, Deletion 1, Replacement 5 and swap 6)

	Original status	Best case
Total cost	312,266	289,290
Cost per sentence	51.11	47.35
Cost per word	2.31	2.14

Table 4. Performance of the original status and of the best case

Table 4 does show some improvement after tuning the disambiguation module. However, the improvement is not apparent. This implies that the disambiguation part is not the major bottleneck for the quality gap. In fact, most translations are ***readable*** and ***understandable*** under human's judgement. So we examined the other parts of the system. We found that the biggest problem is that the translation style does not fit customers' need. We thus conclude that more efforts should be concentrated on the transfer and generation phases, and a transfer and generation model that is capable of adapting the system to different domains and styles {Chan 92b} is required. This case study shows that a quick performance evaluation does play an important role in directing the research direction.

4. Parameterized Feedback Control System Based on the Performance Measure

Through the quick and automatic quantitative distance measure, the system performance can be *on-line* reported in terms of an objective cost function. Therefore, it can be applied in the guided searching in a parameterized, feedback controlled system. The following sections show how the quick performance measure helps to construct such a feedback system. Without a quick performance evaluator, these models will not be made possible.

4.1. Ambiguity Resolution and Lexicon Selection in a Feedback System

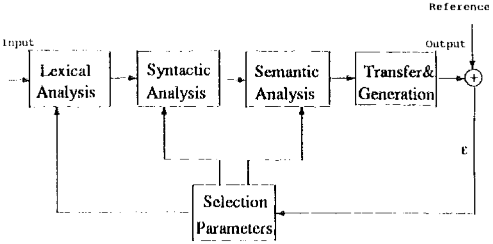

Figure 3. Parameter tuning from feedback

A parameterized feedback-controlled MT system can be modeled as in Figure 3. The control of the system is governed by its static knowledge and a dynamically adjustable parameter set which are used to select the best interpretation among the various ambiguities, or to select the most preferred style in the transfer and generation phases. The probabilistic translation model proposed in [Chen 91] is one such example. In this model, the best analysis is to be selected to maximize an analysis score or *score function* [Chan 92a, Liu 90, Su 88, 89, 91b] of the following form:

$$Score \equiv P\left(Sem_i, Syn_j, Lex_k \mid Words\right)$$

where Sem_i, Syn_j, Lex_k is a particular set of semantic annotation, syntactic structure and lexical category corresponding to some ambiguous construct of the sentence $Words$. Furthermore, the best transfer and generation is to be selected to maximize the following transfer score S_{txf} and generation score S_{gen}:

$$S_{txf} = P\left(T_t \mid T_s\right) \approx P\left(\hat{T}_t \mid \hat{T}_s\right)$$
$$S_{gen} \approx P\left(t \mid \hat{T}_t\right)$$

where T_t, T_s are the target and source of intermediate representations in the form of an annotated syntax tree (AST); $\hat{T}_t$, $\hat{T}_s$ are the normalized version of the AST's, called the normal forms (NF) of the AST's, which are used particularly for transfer and generation, and t is the generated target sentence [Chan 92b].

In such a system, we are to formulate a model as closely to the unknown real language model as possible. Thus, the main task is to estimate the parameters which characterize the model. Because it is not always possible to acquire sufficient data, particularly for complex language models, the estimated parameters might not be able to characterize the real model. Under such circumstances, we can adaptively adjust the estimated parameters according to the error feedback.

In Figure 3, the analysis phase (lexical, syntactic and semantic analyses) of the system is characterized by a set of selection parameters. The input is fed into the lexical analysis phase. The output is generated and acts as the input of transfer phase. In the feedback controlled scheme, a set of revised text, such as the 6,110 Chinese sentences in the previous section, can be used as the reference, and be compared with the translation output of 6,110 sentences. Under the scoring mechanism, the preferred analysis selected may not correspond to the translation with the shortest distance from the reference. Under this circumstance, we can adjust the selection parameters according to the error ϵ (the difference between the reference and the system output) iteratively. Through this *adaptive learning procedure* [Amar 67, Chia 92, Kata 90], the estimated parameters will approach the real parameters very closely. In this way, it will help in ambiguity resolution and lexicon selection. Such a system is made possible to automatically fine-tune the system because the performance measure proposed in this paper provides an on-line response to the iteratively changed parameters.

4.2. Bi-lingual Transfer Model

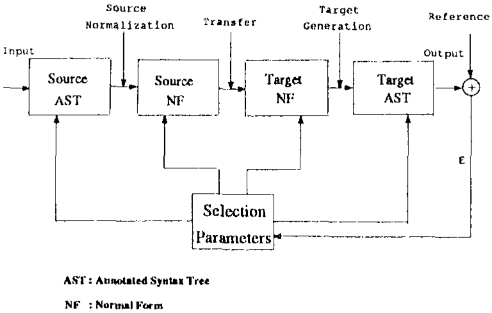

Figure 4. An adaptive learning conceptual model for the transfer and generation phases

As another application of the quick performance measure, we can construct a feedback controlled transfer and generation model. Figure 4 shows such a conceptual model for the parameterized transfer and generation phases [Chan 92b], where AST [Chan 92a] is a syntax tree whose nodes are annotated with syntactic and semantic features and NF is a normalized version of AST, which consists of only atomic transfer units. (See also the previous section for the transfer score and generation score). The Source NF and Target NF are characterized by a set of selection parameters. By jointly considering the parameters characterizing the Source and Target NF, we can adaptively adjust the parameters from the feedback of both directions just like in the previous section. Also the feedback control will make the parameters be tuned to fit the stylistic characteristics of the revised target sentences. Hence, more natural sentences could be generated and less editing effort could be expected in order to get high quality translation. Again, only a quick performance evaluator can make such feedback system practical.

5. Conclusion

The need for performance evaluation is rising, for both customers and system designers. We proposed a performance evaluation method with which system performance can be evaluated automatically and quickly. The approach to improve system performance and feedback controlled MT system is proposed based on such measure. Because the revised text is used directly as reference, the performance measure can indicate real quality gap between users' expectation and system capability.

Though we can not measure the very fine detailed features because there is not very much linguistic knowledge incorporated, our approach has many advantages over conventional approaches. There is no need for human interference. The criteria are consistent and objective. And, we are trying to offer the solutions what users really need. Most important of all, from the feedback of measurement, it is fairly easy for system fine-tuning.

References

[Amar 67] Amari S., 1967. "A Theory of Adaptive Pattern Classifiers," *IEEE Trans. on Electronics Computers*, vol. EC-16, pp. 299–307, June 1967.

[Bell 57] Bellman R.-E., "Dynamic Programming," *Princeton University Press*, 1957, Princeton, NJ, USA.

[Chan 92a] Chang, J.-S., Y.-F. Luo and K.-Y. Su, 1992. "GPSM: A Generalized Probabilistic Semantic Model for Ambiguity Resolution," to appear in *Proceedings of ACL-92*, 30th Annual Meeting of the Association for Computational Linguistics, University of Delaware, Newark, DE, USA, 28 June-2 July, 1992.

[Chan 92b] Chang, J.-S. and K.-Y. Su, 1992. "A Corpus-Based Statistics-Oriented Transfer and Generation Model for Machine Translation," to submit.

[Chen 91] Chen, S.-C., J.-S. Chang, J.-N. Wang and K.-Y. Su, 1991. "ArchTran: A Corpus-based Statistics-oriented English-Chinese Machine Translation System," *Proceedings of Machine Translation Summit III*, pp. 33–40, Washington D.C., USA, July 1–4, 1991.

[Chia 92] Chiang, T.-H., Y.-C. Lin and K.-Y. Su, "Syntactic Ambiguity Resolution Using A Discrimination and Robustness Oriented Adaptive Learning Algorithm," to appear in *Proceedings of COLING-92*, 14th International Conference on Computational Linguistics, Nantes, France, 20–28 July, 1992.

[Devi 82] Devijver, P.A. and J. Kittler. *Pattern Recognition : A Statistical Approach*, Prentice-Hall, London, 1982.

[Kata 90] Katagiri, S. and C.-H. Lee, 1990. "A Generalized Probabilistic Decent Method," *Proc. Acous. Sco. of Japan*, pp. 141–142, Nagoya, Sep. 1990.

[King 90] King, M. and K. Falkedal, 1990. "Using Test Suites in Evaluation of Machine Translation Systems," *Proceedings of COLING-90*, vol. 2, pp. 211–216, the 13th International Conference on Computational Linguistics, Helsinki, Finland, 20–25 Aug. 1991.

[Liu 90] Liu, C.-L., J.-S. Chang and K.-Y. Su, 1990. "The Semantic Score Approach to the Disambiguation of PP Attachment Problem," *Proceedings of ROCLING-III*, pp. 253–270, Taipei, ROC.

[Lowr 75] Lowrence, R. and R.-A. Wagner, 1975. "An Extension of the String-to-String Correction Problem,"*Journal A.C.M.*, vol. 22, no. 2, pp. 177–83, 1975.

[Pigo 82] Pigott, I.-M., 1982. "The Importance of Feedback from Translators in the Development of High-quality Machine Translation," *Practical Experience of Machine Translation*, pp. 61–71.

[Ross 82] Rossi F., 1982. "The Impact of Posteditors' Feedback on the Quality of MT," *Practical Experience of Machine Translation*, pp. 113–117.

[Su 88] Su, K.-Y. and J.-S. Chang, 1988. "Semantic and Syntactic Aspects of Score Function," *Proceedings of COLING-88*, vol. 2, pp. 642–644, 12th International Conference on Computational Linguistics, Budapest, Hungary, August 22–27, 1988.

[Su 89] Su, K.-Y., J.-N., Wang, M.-H. Su, and J.-S. Chang, 1989. "A Sequential Truncation Parsing Algorithm Based on the Score Function," *Proceedings of International Workshop on Parsing Technologies(IWPT 89)*, pp. 95–104, Pittsburgh, PA, USA.

[Su 90] Su, K.-Y., and J.-S. Chang, 1990. "Some Key Issues in Designing MT Systems," *Machine Translation*, vol. 5, no. 4, pp. 265–300, 1990.

[Su 91a] Su, K.-Y. and C.-H. Lee, 1991. "Robustness and Discrimination Oriented Speech Recognition Using Weighting HMM and Subspace Projection Approaches," *Proceedings of IEEE ICASSP-91*, Toronto, Ontario, Canada.

[Su 91b] Su, K.-Y., J.-N. Wang, M.-H. Su, and J.-S. Chang, 1991. "GLR Parsing with Scoring," M. Tomita (ed.), *Generalized LR Parsing*, Kluwer Academic Publishers.

[Wagn 74] Wagner, C.-K. and M.-J. Fischer, 1974. "The String-to-String Correction Problem," *Journal A.C.M.*, vol. 21, no. 1, pp. 168–173.

[Wu 91] Wu., M.-W., J.-S. Chang and K.-Y. Su, 1991. "The Current Status of ArchTran: A Corpus-Based Statistics-Oriented English-Chinese Machine Translation System," *Proceedings of the 1991 Workshop on Machine Translation*, pp. 123–138, Nantou, Taiwan, ROC, 24–26 June, 1991.

[Yu 91] Yu, S., 1991. "Automatic Evaluation of Output Quality for Machine Translation System," *International Symposium on Multilingual Machine Translation*, Beijing, China, pp. 57–58, 19–21 Aug. 1991.