Confidence Estimation for Machine Translation

John Blatz
Princeton
jblatz@princeton.edu

Erin Fitzgerald
Johns Hopkins
erinf@jhu.edu

George Foster
LTRC, Canada
george.foster@nrc.gc.ca

Simona Gandrabur
University of Montreal
gandrabu@iro.umontreal.ca

Cyril Goutte
XRCE, France
cyril.goutte@xrce.xerox.com

Alex Kulesza
Harvard
kulesza@fas.harvard.edu

Alberto Sanchis
UPV, Spain
asanchis@iti.upv.es

Nicola Ueffing
RWTH Aachen
ueffing@cs.rwth-aachen.de

Abstract

We present a detailed study of confidence estimation for machine translation. Various methods for determining whether MT output is correct are investigated, for both whole sentences and words. Since the notion of correctness is not intuitively clear in this context, different ways of defining it are proposed. We present results on data from the NIST 2003 Chinese-to-English MT evaluation.

1 Introduction

All current NLP technologies make mistakes. Applications built on these technologies can cope with mistakes better if they have some reliable indication of when they may have occurred. For instance, in a speech recognition dialog system, low confidence in the analysis of a user's utterance can lead the system to prompt for a repetition. This strategy has the potential to significantly improve the system's usability if accurate estimates of correctness can be made.

The binary classification problem of assessing the correctness of an NLP system's output is known as confidence estimation (CE). It has been extensively studied for speech recognition, but is not well known in other areas. The motivation for our work was to apply CE techniques to another NLP problem, measure performance, and attempt to draw general conclusions. We focused on machine translation because it is an important area of NLP, and one where CE has the potential to enable new applications.

In this paper we study confidence estimation for both sentences and words in MT output. Since most MT systems operate at the sentence level, sentences are natural targets for correctness judgements. The main challenges in making these judgements are that MT output is rarely correct at the sentence level to begin with, and that there is no satisfactory automatic method for determining whether or not a given output sentence *is* correct, even if reference translation(s) are available. Applications for sentence-level CE include filtering translations for human post-editing or information gathering, combining output from different MT systems, and active learning (Ngai and Yarowsky, 2000).

CE for words is relatively unaffected by the problems that apply at the sentence level. Individual words are more likely to be correct than are whole sentences, and their correctess can be assessed fairly reliably by comparison to reference translations. On the other hand, what correctness means is less obvious at this level; a word could be correct in *some* possible translation, but wrong in the current context. Potential applications here include post-editing, interactive machine translation systems, recombination of multiple sentence-level MT hypotheses, and improved search algorithms (Neti et al., 1997).

In the remainder of the paper, we first give some background on CE in general (s2), then describe our experimental setting (s3), present sentence-level (s4) and word-level (s5) results, and make some concluding remarks (s6).

2 Background

The goal of CE is to characterize the behaviour of a base NLP system that produces an output y given an input x. One way of doing so, which we call *weak CE*, is to build a classifier that takes x and y as input and returns a correctness score. Various decisions can then be based on (suitably optimized) thresholding against this score. When scores are direct estimates of correctness probabilities, they have a somewhat wider range of applications; we refer to this as *strong CE*. Both strong and weak approaches are reported in the speech literature; evaluation techniques for each are described in section 3.3 and in (Siu and Gish, 1999).

CE techniques also differ in whether or not they use a separate "CE layer" distinct from

corpus	src	refs	sent	word
NIST01	700	4	train	train
NIST01	293	4	train	val
LDC1	4107	1 or 4	train	
LDC2	565	4	val	
NIST02	878	4	test	test

Table 1: Corpora. Columns give number of source sentences and reference translations, and the split used for sentence and word experiments.

the base NLP system. Many approaches, eg (Wessel et al., 2001), derive confidence scores, such as posterior probabilities $P(y|x)$, directly from quantities in the base system. However, methods in which the CE portion is separate predominate. Although these have the disadvantage of requiring a training corpus of examples labelled for correctness, they are more powerful and modular. A wide range of machine learning algorithms have been tried in this setting, including naive Bayes (Sanchis et al., 2003), neural nets (Guillevic et al., 2002), and SVMs (Zhang and Rudnicky, 2001).

All previous work on CE for MT has been done by some of us. Ueffing et al (Ueffing et al., 2003) describe several direct methods, including posterior probabilities, for estimating the correctness of individual words in MT output. Gandrabur and Foster (Gandrabur and Foster, 2003) describe the use of a neural-net CE layer to sharpen probability estimates for text predictions in an interactive translators' tool.

3 Experimental Setting

3.1 Corpora

Our corpora consist of Chinese-to-English evaluation sets from NIST MT competitions, as well as a large multi-reference corpus provided by the Linguistic Data Consortium (LDC), cf. table 1. These were divided into separate train, validation, and test portions. We obtained ouput from the ISI Alignment Template MT system (Och and Ney, 2004) that participated in the 2003 NIST evaluation (NIST, 2003). For each source sentence, the system produced an N-best list of translation candidates, of which we used the top 1,000 for all experiments described in this paper. Each resulting source-sentence/target-candidate pair was treated as an independent example (many examples for word experiments), whose correctness was established from the reference translation(s) available for the source sentence. The exact method of defining correctness varied across different experiments, as described below.

3.2 CE Techniques

Our data can be viewed as a collection of pairs ($\mathbf{x}$,c) in which $\mathbf{x}$ is a feature vector and c a correctness indicator. We explored different ways of capturing the relationship between $\mathbf{x}$ and c. For weak CE, we used scores derived directly from $\mathbf{x}$, and also (at the sentence level) multilayer perceptron (MLP) based regression models of MT evaluation scores from which c was derived. For strong CE, we used naive Bayes and MLP to estimate the probability of correctness $P(c = 1|\mathbf{x})$. Our choice of these learning methods was driven by constraints on the resources required for training on millions of examples: naive Bayes requires only a single pass over the data for parameter estimation; while MLP typically requires only a few passes when using stochastic gradient descent.

Naive Bayes (NB)

In a probabilistic setting, the posterior class probability is given by $P(c|\mathbf{x}) \propto P(c)P(\mathbf{x}|c)$. The *Naive Bayes* assumption is that features are statistically independent: $P(\mathbf{x}|c) = \prod_{d=1}^{D} P(x_d|c)$, where D is the dimension of the feature vector $\mathbf{x}$. Parameters are estimated using an *absolute discounting smoothing* of the maximum likelihood solution. A small constant $b \in (0, 1)$ is discounted from every positive count and distributed accross all events with null counts. Denoting N and $N(c)$ the number of examples in total and in class c, respectively, $N(x_d, c)$ the number of examples in class c with feature value x_d, and N_+, N_- the number of possible values of x_d with $N(x_d, c) > 0$ and with $N(x_d, c) = 0$, respectively, estimates are $P(c) = N(c)/N$ and:

$$P(x_d|c) = \begin{cases} \dfrac{N(x_d, c) - b}{N(c)} & \text{if } N(x_d, c) > 0 \\ \dfrac{b}{N(c)} \dfrac{N_+}{N_-} & \text{if } N(x_d, c) = 0 \end{cases}$$

For word-level CE, the word class prior probability is considered in the word posterior class estimation: $P(c|\mathbf{x}, w) \propto P(c|w)P(\mathbf{x}|c)$, where $P(c|w)$ is smoothed as above.

Continuous features are discretised into a fixed number of bins (usually 20) by visual inspection of the histograms of the feature values.

Multi Layer Perceptron

Multi Layer Perceptrons implement a non-linear mapping of the input features by combining

layers of linear transformation and non-linear transfer functions (Bishop, 1995). Parameter are estimated by minimising a squared error loss for regression and the negative log-likelihood for classification. In our experiments, we used a stochastic gradient descent, because it tends to be quite efficient on redundant data.

As examples typically arise from the same source sentence, they have many similar features and are therefore highly redundant. Convergence of stochastic gradient descent is guaranteed under certain conditions, in particular examples must be presented in random order. In our case, we usually have too many examples to first load the training set in memory, then pick examples at random. We therefore implemented a random caching mechanism, where data are loaded sequentially but unloaded randomly, in order to simulate an independant random pick from the entire training set. Although the examples are only approximately independant using this caching system, we observed empirically that when the cache was large enough to hold all the examples corresponding the several source sentences, the final performance was indistinguishable from a model trained using truly independent random examples. This random cache was implemented in the framework of Torch (Collobert et al., 2002).

3.3 Metrics for Evaluation

As mentioned earlier, we are interested in assessing the performance of CE techniques in two settings: strong CE, requiring accurate probabilities of correctness; and weak CE, requiring only binary classification. In order to evaluate our models, we use different metrics, all calculated over a test set $\{(\mathbf{x}^{(i)}, c^{(i)})\}_{i=1...n}$, where the indicator $c^{(i)}$ is 1 iff $\mathbf{x}^{(i)}$ is correct, and 0 otherwise. We let n_1 and n_0 designate the numbers of correct and incorrect examples.

Strong CE metric: NCE

A standard way of measuring the fit between a probabilistic model and a test corpus is negative log-likelihood (or cross entropy): $\text{NLL} = -\sum_i \log P(c^{(i)}|\mathbf{x}^{(i)})/n$. To remove dependence on the proportion of correct examples in the corpus, we use *normalized* cross entropy (NCE):

$$\text{NCE} = (\text{NLL}_b - \text{NLL})/\text{NLL}_b \qquad (1)$$

The baseline NLL corresponds to assigning fixed probabilities of correctness based on the empirical class frequencies: $\text{NLL}_b = -(n_0/n)\log(n_0/n) - (n_1/n)\log(n_1/n)$.

Weak CE metrics: CER and (I)ROC

The metrics we use for weak CE attempt to capture the discriminability of the classification function across the range of all thresholds used to decide correctness.

The simplest metric is the classification error rate (CER): the proportion of examples on which the classifier's output differs from the true correctness indicator. The values of CER we use are based on thresholds optimized on the test set (for sentence-level experiments), and on the validation set (for word experiments). The baseline is a classifier which assigns all examples to the most frequent class, for which $\text{CER}_b = \min(n_0, n_1)/n$.

Another common way to assess the discriminability of a classifier is to use the receiver operating characteristic (ROC) curves (Duda et al., 2001). These plot correct-reject ratio (true negatives/n_0) vs correct-accept ratio (true positives/n_1) for different thresholds. The ROC curve lies in the unit square, with random choice corresponding to the diagonal and perfect discrimination corresponding to the edges. A related quantitative measure is the area under the ROC curve or IROC, which gives a global indication of the discriminability over all possible rejection thresholds.

4 Sentence-Level Experiments

In order to assign a correctness c to each translation hypothesis, we threshold automatic MT evaluation measures. We use the two measures which correlated best with human judgement in the evaluation exercise described in (Blatz et al., 2003):

WERg, the word error rate, normalised by the length of the Levenshtein alignment; and

NIST, the sentence-level NIST score (Doddington, 2002).

We use two different thresholds for each score, giving four problem settings in total. The first threshold produces 5% of "correct" examples, and is intended to be sufficient for gisting purposes. The second one tags 30% of examples as correct, which we believe would be enough for applications that require a simple bag-of-words translation, such as cross-language IR.

4.1 Features

We used a total of 91 sentence-level features, which we summarize briefly in this section. A detailed list is given in (Blatz et al., 2003).

Base-Model Intrinsic

As described in (Och and Ney, 2004), the ISI MT system is based on a maximum entropy model defined over twelve feature functions; we used the values returned by these functions as features in our CE models. Another class of features captures various pruning statistics from the base model's search algorithm.

N-best List

A large set of features were calculated over the N-best list generated for each source sentence. This includes simple statistics such as rank of the current hypothesis, ratio of its score to the best score, average length of hypotheses in the list, and ratio of total number of words to source-sentence length. Additionally, various features were based on the *center hypothesis* having minimal average Levenshtein distance to all others.

Source Sentence

Features intended to capture the translation difficulty of the current source sentence included its length, various ngram frequency statistics, and trigram language model scores.

Target Sentence

Similar to the above, we used features to capture the viability of the current target hypothesis, including external and N-best-based (word and phrase) language model scores, word frequencies over the N-best list, and simple parenthesis and quotation matching.

Source/Target Correspondence

Features in this class aimed to capture the translation relation between the source sentence and target hypothesis. These included IBM model 1 (Brown et al., 1993) probabilities in both directions, word-alignment monotonicity, and various kinds of agreement with other word-level translations in the N-best list, including a semantic similarity metric based on WordNet.

4.2 MLP Experiments

We compared various MLPs, trained on all features, on the four problem settings described above. Models used varying numbers of hidden units (from 0 to 20), and either classification or regression.

Table 2 shows the performance of the best configurations for classification and regression. The number of hidden units for each model is omitted because it has no clear correlation with performance. However, there *is* a clear trend in which classification models do better than regression models, particularly as measured by IROC. (This is not completely surprising, given that classification MLPs were specifically trained for the corresponding thresholds.) Globally, performance is better than the baseline in all cases except CER for the NIST 5% setting.

setting	IROC	CER	NCE
N 05%	.800 ± .002	3.21 ± .03	9.18 ± .61
N 30%	.763 ± .001	27.10 ± .10	15.62 ± .09
W 05%	.818 ± .002	5.57 ± .05	16.94 ± .21
W 30%	.734 ± .001	28.95 ± .06	12.09 ± .10
N 05%	.757 ± .002	3.10 ± .03	
N 30%	.762 ± .001	27.20 ± .10	
W 05%	.741 ± .002	5.54 ± .05	
W 30%	.723 ± .001	29.04 ± .04	

Table 2: Best results for classification (top box) and regression, with 95% confidence bounds. N and W stand for NIST and WERg in the problem setting column. Baseline values for CER are 3.21%, 32.5%, 5.65%, and 32.5% for each problem setting, in order. Note that the results on each line in this table are not necessarily generated by a single model.

4.3 Feature Comparison

We compared the contributions of all features, both as individual confidence scores and as part of feature groups used to train MLPs. To group features, we classified them in two independent ways according to whether they apply to the source sentence, the target hypothesis, or both; and according to whether they depend on the base model or could be calculated without knowledge of it. We also treated the base-model's scores as group on their own. All feature experiments were performed only for the NIST 30% setting.

Results are shown in table 3 and figure 1. The most striking observation is that the MLP trained on only the twelve feature functions from the base model is almost as good as the one trained on all features. Another pattern is that features that depend on the base model are more useful that those that do not, and features that apply to the target hypothesis are more useful than ones that apply only to the source sentence (as well as, to a much lesser extent, those that apply to a source/target pair). A final conclusion is that a model that has been trained on labelled data—regardless of the feature set used—is better at discriminating than any single feature on its own.

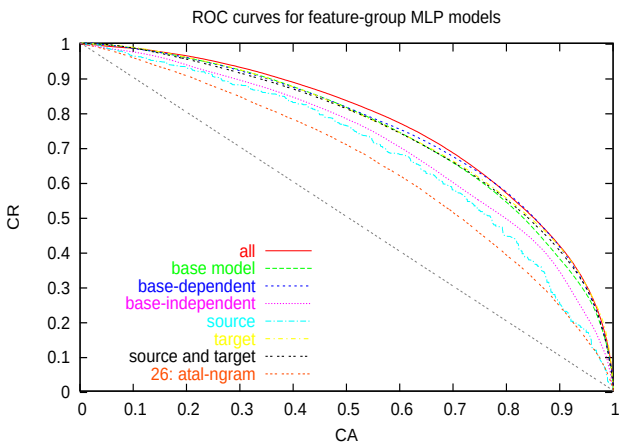

Figure 1: ROC curves for models trained on different feature groups. Note that the "wiggly" appearance of the curve for the *source* group is due to the fact that these features are invariant over all entries in a given N-best list, which leads the model to assign large blocks of examples exactly the same probability of correctness.

G	IROC	CER	NCE
All	.763 ± .001	27.10 ± .10	15.62 ± .09
Base	.746 ± .001	27.82 ± .09	13.97 ± .08
BD	.754 ± .001	28.02 ± .10	14.83 ± .10
BI	.712 ± .001	29.88 ± .09	9.76 ± .09
S	.687 ± .002	30.41 ± .07	7.23 ± .08
T	.751 ± .001	27.91 ± .08	14.48 ± .09
ST	.746 ± .001	28.47 ± .10	13.62 ± .07
S1	.648 ± .001	32.14 ± .08	

Table 3: Results for models trained on different feature groups. All is all features, Base is base model scores; BD and BI are base-model dependent and independent; S, T, and ST are source, target, and both; and S1 is the best single feature.

5 Word-Level Experiments

It is not intuitively clear how to classify words in MT output as correct or incorrect when comparing the translation to one or several references. We implemented a number of different measures that were inspired by automatic evaluation metrics like WER and PER.

Pos: This error measure considers a word as correct if it occurs in exactly this target position in the reference translation.

WER: A word is counted as correct if it is Levenshtein-aligned to itself in the reference.

PER: A word is tagged as correct if it occurs in the reference translation. The word order is completely disregarded, but the number of occurrences is taken into account.

For all error metrics, we determine the reference with minimum distance to the hypothesis according to the metric under consideration and classify the words as correct or incorrect with respect to this reference.

These error metrics behave significantly different with regard to the percentage of words that are labeled as correct. Pos is very pessimistic with only 15% correct words on the corpora described in section 3.1, whereas WER labels 43% as correct, and PER 64%, respectively. Note that those figures are not the translation errors for the system output. They are calculated for every hypothesis in the N-best list (and not only for the single best translation).

5.1 Features

We used 17 features in total which we will describe shortly in this section. For more details, see (Blatz et al., 2003).

SMT Model Based Features

We investigated two features that are based directly on namely the Alignment Template MT model (Och and Ney, 2004). One gives the identity of the so-called Alignment Template, i.e. the bilingual phrase, that was applied in the translation of the current target word. Another feature specifies whether the target word was translated by a rule based system or not. This rule based system was integrated into the translation process for the translation of special phenomena such as dates and time expressions.

IBM Model 1

We implemented one feature that determines the average translation probability of the target word e over the source sentence words according to Model1 introduced by IBM in (Brown et al., 1993). This captures a sort of topic or semantic coherence in translations.

Word Posterior Probabilities and Related Measures

We investigated three different features introduced in (Ueffing et al., 2003) that are calculated rather similarly: relative frequencies, rank weighted frequencies and word posterior probabilities. Each of them is based on determining those sentences in the N-best list that contain the word e under consideration in a certain position. The first variant (called *any* in table 4) regards all those sentences that contain e at all, whereas the second variant (*source*) considers all sentences where e occurs as the translation of the same source word(s). The third variant

(*target*) determines only those target sentences containing e in exactly the same position.

Target Language Based Features

We implemented three different features using the semantic data provided by WordNet. The first similarity feature is the average semantic similarity from the word in question to the word aligned to the same source position in each of the top three hypotheses. The two other features come from WordNet's polysemy count, for details see (Blatz et al., 2003).

Two more target language based features were implemented. One is a basic syntax check that looks to highlight hypotheses with mismatched parentheses and/or quotation marks. The second feature counts the number of occurences in the sentence for each word in the target sentence.

5.2 Performance of Single Features

Using the Naive Bayes classifier, we tested the performance of single features for word confidence estimation. Additionally, we combined the best 3 and all 17 features with the same classifier. Table 4 shows the confidence estimation performance of single features in terms of CER and IROC using the error measure PER for labeling words as correct or incorrect. The features which yield the best results are the word posterior probability, rank weighted frequency, and relative frequency (WPP) with respect to occurrence of the word in any position in the target sentence. Those three features give a significant improvement over the baseline of more than 5% absolute in CER. The feature based on Model1 also discriminates very well, followed closely by the WPP with regard to the aligned source position(s).
The combination of three of the best performing features (word posterior probabilities with respect to different criteria and the Model1 based feature) yields a significant improvement over the performance of any of the single features. There is no significant change in CER or IROC if more information is added by combining all 17 features.

5.3 Comparison of Different Models

For word level confidence estimation, we investigated several different MLP architectures, with the number of hidden units ranging from 0 to 20.

Figure 2 compares the tagging performance for different MLP architectures and for the Naive Bayes classifier, including all features.

Feature	CER	IROC
Baseline	36.2	–
WPP any	30.8–30.9	.706–.707
Model1	31.2	.699
WPP source	31.9	.694–.695
WPP target	32.5–32.7	.686–.689
SMT model based	33.1	.671–.673
Target based	33.1–33.4	.666-.668
top 3	29.2	.733
All	29.6	.736

Table 4: CER [%] and IROC for single features and their combination using Naive Bayes; Word Error Measure: PER.

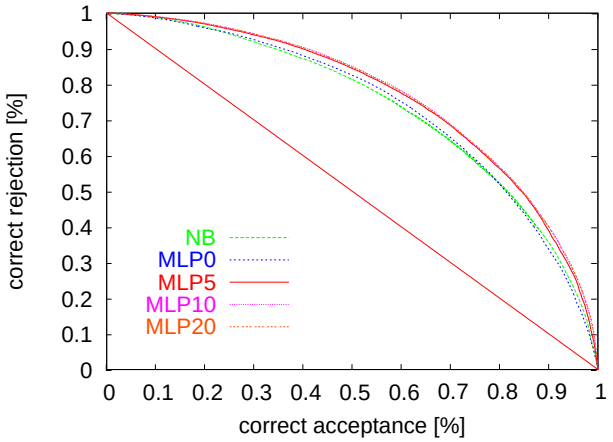

Figure 2: ROC curves for PER, MLPs with different numbers of hidden units and Naive Bayes, combining all features.

We see that the Naive Bayes classifier and the MLP with zero hidden units have a very similar performance. But as soon as the MLP gets more complex by the addition of more hidden units, the MLP outperforms the Naive Bayes approach significantly. There is no significant difference between the MLPs consisting of 5 to 20 hidden units.

5.4 Comparison of Word Error Measures

Table 5 compares the IROC values for word confidence estimation using an MLP with 20 hidden units for all the error measures described at the

Error Measure	Pos	WER	PER
IROC	.734	.703	.766

Table 5: Comparison of IROC values of an MLP with 20 hidden units for different error measures (all features).

beginning of the section. We see that classification according to some of the error measures is easier to learn than according to others. The highest discriminability is obtained for the most "relaxed" measure PER, followed by Pos and WER results are slighly worse.

6 Conclusion

We have reported on the use of various techniques for classifying MT output as correct or not, at both the sentence and word levels. Both levels present problems for the definition of correctness. At the sentence level we resolved these problems by using automatic MT evaluation metrics and re-defining "correctness" to be above a certain threshold (of match to reference translations), which we feel should correspond to usability within different applications. At the word level we investigated three strategies, differing in strictness, for matching corresponding words in reference translations.

Our main conclusions can be summarized as follows:

- Training a separate layer using machine-learning techniques is better than relying solely on base model scores.
- Features derived from the base model are more valuable than external ones, and should be tried first before investing effort in the implementation of complex external functions.
- Features based on N-best lists are more valuable than ones based solely on individual hypotheses.
- Features that capture properties of the target text are more valuable than those that do not.
- Multi-layer perceptrons (neural nets) outperform naive Bayes models. MLPs with more hidden units can give better performance than those with fewer.

In future work, we look forward to using the techniques developed here within various applications described in the introduction. We also intend to continue to refine our definitions of correctness to make them more stable and more broadly applicable.

Acknowledgement

This material is based upon work supported by the National Science Foundation under Grant No. 0121285.

References

C.M. Bishop. 1995. *Neural Networks for Pattern Recognition.* Oxford.

J. Blatz, E. Fitzgerald, G. Foster, S. Gandrabur, C. Goutte, A. Kulesza, A. Sanchis, and N. Ueffing. 2003. Confidence estimation for machine translation. Final report, JHU / CLSP Summer Workshop.

P.F. Brown, S.A. Della Pietra, V.J. Della Pietra, and R.L. Mercer. 1993. The mathematics of Machine Translation: Parameter estimation. *Computational Linguistics*, 19(2):263–312.

R. Collobert, S. Bengio, and J. Mariéthoz. 2002. Torch: a modular machine learning software library. Technical Report IDIAP-RR 02-46, IDIAP.

G. Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proc. HLT-02.*

R.O. Duda, P.E. Hart, and D.G. Stork. 2001. *Pattern Classification.* Wiley.

S. Gandrabur and G. Foster. 2003. Confidence estimation for text prediction. In *Proc. CoNLL03.*

D. Guillevic, S. Gandrabur, and Y. Normandin. 2002. Robust semantic confidence scoring. In *Proc. ICSLP'02.*

C. Neti, S. Roukos, and E. Eide. 1997. Word-based confidence measures as a guide for stack search in speech recognition. In *ICASSP'97.*

G. Ngai and D. Yarowsky. 2000. Rule writing or annotation: Cost-efficient resource usage for base noun phrase chunking. In *Proc. ACL'00.*

NIST, editor. 2003. *Proc. NIST Workshop on Machine Translation Evaluation.*

F.J. Och and H. Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics.* To appear.

A. Sanchis, A. Juan, and E. Vidal. 2003. Improving utterance verification using a smoothed naive Bayes model. In *ICASSP'03.*

M. Siu and H. Gish. 1999. Evaluation of word confidence for speech recognition systems. *Computer Speech and Language*, 13(4).

N. Ueffing, K. Macherey, and H. Ney. 2003. Confidence measures for Statistical Machine Translation. In *Proc. MT Summit IX.*

F. Wessel, R. Schlüter, K. Macherey, and H. Ney. 2001. Confidence measures for large vocabulary continuous speech recognition. *IEEE Tr. Speech Audio Proc.*, 9(3).

R. Zhang and A. Rudnicky. 2001. Word level confidence annotation using combinations of features. In *Eurospeech*, pages 2105–2108.