Improving Speech Recognition Accuracy with Multi-Confidence Thresholding

Shuangyu Chang

Tellme Networks, Inc.
1310 Villa Street, Mountain View, CA 94041, USA
shawn@tellme.com

10.21437/Interspeech.2006-450

Abstract

Confidence-based thresholding plays an important role in practical speech recognition applications. Most previous works have focused on directly improving confidence estimation within the recognition engine. A complementary approach that does not require access to recognizer internal is to optimize confidence threshold settings. This paper describes a general multi-confidence thresholding algorithm that automatically learns different confidence thresholds for different utterances, based on discreet or continuous features associated with a speech utterance. The algorithm can be applied to any speech recognition engine with a confidence output. A learned multi-threshold setting is guaranteed to perform at least as well as a baseline single-threshold system on training data. A significant improvement on overall accuracy can often be obtained on test data, as demonstrated with experiments on two real-world applications.

Index Terms: speech recognition, confidence threshold, IVR system, pattern recognition

1. Introduction

In practical Interactive Voice Response (IVR) systems, there are usually two major components returned by the speech recognition engine. One is the textual transcription and semantic interpretation of a user speech utterance; the other one, often less noticed by individual users but no less important than the former, is the confidence measure on the recognition result. The IVR system compares the confidence measure with a predetermined confidence threshold and only accepts the recognition result if the confidence is above the threshold. Thus an accurate confidence estimation and a properly set confidence rejection threshold can significantly improve the tradeoff between minimizing false acceptance (*FA*) of erroneous recognition results and maximizing correct acceptance (*CA*) of good recognition results.

The importance of confidence-based thresholding in practical applications has prompted much attention and research effort in the speech recognition community. One of the metrics for comparing various speech recognition systems in NIST sponsored evaluations is indeed the quality of confidence estimation [1]. There has been an abundance of work on confidence estimation in the literature. A common theme among many of these works was a direct improvement of confidence measure computation within the speech recognition engine (e.g. [2][3][4]). These works generally require access to the nuts and bolts of the speech recognition engine, making it difficult for applications to adapt to characteristic differences of data from different tasks. A complementary approach is to focus on improving the confidence rejection threshold settings. However, researches along this approach are relatively scarce, and previous works generally involve *ad hoc* methods for specific pattern recognition applications (e.g., [5][6]).

We propose a general multi-confidence thresholding framework to improve the overall accuracy of a speech recognition system. Unlike the conventional single-threshold framework where the same confidence threshold is applied on all speech utterances of a given task, the multi-threshold framework employs several different confidence thresholds, each applicable only to a subset of utterances depending on one or more features associated with an utterance. The key idea is to model the systematic inaccuracies in confidence estimation of the underlying speech recognition system, and to use this knowledge to improve threshold settings. As described below, the proposed method is applicable to any speech recognition engine without an access to recognizer internals. It can be adapted to improve accuracy of different speech recognition tasks while respecting task-specific *FA-CA* tradeoffs.

The next section gives a detailed description of the multi-confidence thresholding framework, including algorithms for automatically learning parameters from training data for both discrete and continuous features. Experimental results are then presented on two real-world speech recognition tasks, followed by a discussion and conclusions.

2. Multi-Confidence Thresholding

Assume for a particular speech recognition task, we have a fixed set of recognition grammar and speech recognition engine, which may include a semantic interpretation component. For each speech utterance, the recognition engine processes the audio input and generates an output of recognized text (and possibly its semantic interpretation), as well as a confidence score. For ease of discussion, let us focus on only the one-best result from the recognizer.

The task of confidence thresholding is to determine whether or not the system should accept a recognition result, in order to optimize the expected value of some particular accuracy metric. For specific applications our goal is often to maximize the correct acceptance rate over all utterances (*CA/all*) while keeping the false acceptance rate (*FA/all*) at some low level. In live applications the proposed multi-threshold framework classifies each utterance into one of several partitions based on some features associated with the utterance, and a different threshold is applied to utterances in each partition.

In the following descriptions we assume a training data set of utterances and their recognition/interpretation results are available, including recognition confidence scores. Reference text/interpretation for the same utterances should also be

available for performance evaluation.

2.1. Feature determination

The first step to set up a multi-threshold system is to find suitable features for partitioning speech utterances. It is desirable to have utterances in each partition possess the same type of confidence estimation error from the recognizer. For example, if the recognizer systematically underestimates the confidence for shorter utterances but overestimates for longer ones, the utterance duration would be a good partitioning feature. It is important that the feature can be derived from available information related to the utterance.

To determine partitioning rules for a given feature, it is useful to consider a feature as either discrete or continuous. This distinction is made because different methods can be applied to find partitions. For a discrete feature, such as speaker gender (*male vs. female*), utterances can be partitioned directly based on the label of a category. In some cases the number of distinct categories is too large to be used directly and some categories need to be combined. Various automatic clustering algorithms can help this process [7], but due to space limitation we omit further discussions here. Some other examples of discrete features include semantic interpretation class, estimates of speaker characteristics such as age, emotion, or dialect, and time-of-day (e.g. *business hour vs. after hour*) of the call.

For a continuous feature, however, it is necessary to find boundary values to divide the feature value range into separate partitions suitable for multi-threshold settings. An automatic algorithm for finding boundary values is described in Section 2.3. Besides utterance duration some other examples of continuous features include utterance noise estimate, number of words in recognition result, recognition latency, and prior recognition rate of the same caller.

In following sections, for ease of discussion let us focus on the case when only one feature is used for partitioning. Extension to multiple features is discussed in Section 4.

2.2. Computing optimal thresholds

If a feature and associated partitioning rules are already identified, it is straightforward to compute the optimal confidence threshold for each partition, in order to maximize the *CA/all* across all partitions while keeping *FA/all* below a certain level. For each partition *p*, we first compute the *CA* rate *(CA/p)* and *FA* rate *(FA/p)* over all utterances in that partition, at each confidence level. Then, we compute the *CA/all* and *FA/all* rates for each combination of confidence thresholds among all partitions and find the one that has the maximum *CA/all* rate with *FA/all* rate below our target level.

Exhaustively searching for optimal threshold combination generally works well. However, the size of the search space grows exponentially as the number of partitions and in some cases an exhaustive search becomes intractable. Heuristic methods such as simulated annealing and gradient decent with random restart can significantly speed up the search [7]. Empirically we found that heuristic search results are usually as good as or very close to the exhaustive search result.

2.3. Automatic partitioning of continuous feature

As described above, a continuous feature requires dividing the range of feature value into partitions. An exhaustive search to find an optimal set of boundary values can be intractable for even a modest feature value range. Here we propose a greedy, iterative partitioning algorithm that is fairly efficient and is able to find near-optimal solutions in most cases.

The algorithm first finds the best binary partitioning of the entire feature range by testing each possible boundary value. The key to a successful binary partitioning is a proper definition of the goodness of a boundary value. The goal is to find two partitions, when each assigned a different threshold, provide a combined *CA/all* better than if a single threshold is used, along with a similar or lower combined *FA/all*. One way to quickly assess a binary partitioning is to compare the slopes of the *CA-FA* curves at the single threshold we would have used (see Figure 1 for an example). To keep the combined *FA/all* level in-check, the confidence thresholds must move in opposite directions along the two *CA-FA* curves. If the slopes are very similar, the combined *CA/all* and *FA/all* changes due to movements on the *CA-FA* curves (i.e. changing confidence thresholds) would roughly cancel each other out, thus providing little or no net gain in overall performance. On the other hand, if the two slopes are fairly distinct, movements along the two curves would generate a better combined *CA/all* rate for some target *FA/all* rate, than a single threshold. This insight leads us to define the goodness measure of a boundary value as the slope difference between the *CA-FA* curves on the two resulting partitions, at the previously optimal single confidence threshold.

This binary partitioning process is iterated on the two resulting partitions until some stopping criteria are met. Useful stopping criteria can include reaching maximum number of partitions, reaching minimum number of utterances remaining in each partition, or no significant slope difference achieved with new partitions. Not only do these stopping criteria reduce unnecessary computational resource usage, they also help reduce the risk of overtraining.

2.4. Pseudo code of entire algorithm

Putting it all together below is a pseudo code outlining the complete algorithm for setting multiple confidence thresholds on a given set of training data, *T*. Again, we focus on the case of a single partitioning feature. We assume there is a target *FA/all* rate that should not be exceeded.

I. Finding Partitioning Rules
   a. For discrete feature:
      i. Put each training utterance into a matching partition based on the feature label
      ii. Put all remaining training utterances into another partition
   b. For continuous feature:
      i. Not to exceed the target *FA/all* rate, find the single confidence threshold *C* that maximizes *CA/all* on training set *T*.
      ii. Set the entire training data set *T* as the active partition, *A*.
      iii. For each possible boundary value
         1. Separate *A* into two sub-partitions: *P1, P2*
         2. Compute the local slope for the *CA-FA* curve of each of *P1* and *P2* at *C*
         3. Compute the slope difference *d*
      iv. Find the boundary value that maximizes *d* and satisfies all partitioning requirement such as minimum size of each partition
      v. Replace *A* with the two new partitions, obtained by

dividing *A* with the boundary value found above
vi. Among all partitions find the largest untried partition and set it as the active partition *A*
vii. Loop back to (iii) above until a stopping criterion is met

II. Setting Confidence Thresholds
a. For each partition *P* found above
i. Compute *CA/P* and *FA/P* for each confidence threshold level in consideration
ii. Save these results and the size of *P*
b. For each combination of thresholds of all partitions
i. Compute combined *CA/all* and *FA/all* using results saved in (a)
c. Find the maximum combined *CA/all* not exceeding target *FA/all*. (b) and (c) can be sped up by using a heuristic search such as simulated annealing
d. (Optionally) Find the maximum combined *CA/all* for each *FA/all* level to yield a combined *CA-FA* curve

3. Experiments

This section presents experimental results of applying multi-confidence thresholding to two different practical applications, both involving speaker-independent speech recognition over telephone. The first uses a continuous feature and the second uses a discrete feature.

3.1. Continuous feature example

The first experiment is a listing recognition task from an automated business name search application. Callers from a particular locality are prompted to say the name of a business listing and the system attempts to recognize the caller's utterance against a grammar containing potentially hundreds of thousands listings in the locality. The system needs to decide whether or not to accept a recognition result based on recognizer output, as well as the associated confidence measure. If the result is accepted, the system plays back some information related to the recognized result (e.g. phone number); otherwise, the system plays a different message and re-prompts the caller, or transfers the caller to a human operator for further assistance. Generally we would like to maximize *CA/all*, which correlates with the overall automation rate, while minimizing *FA/all*, which impacts the caller satisfaction with the service.

The baseline system employs a single confidence threshold for all utterances, tuned to provide certain level of *FA/all* that is deemed acceptable for this task. In experiments, a multi-confidence threshold setting was trained automatically using the proposed algorithm with a numeric feature, the duration of utterance. This research data set consists of *6,988* listing utterances from mostly different callers in one large US city. We randomly divided the data into two equal-sized data sets and performed a two-fold cross validation. That is, we first performed training on the first set and tested on the second set, and then trained on the second set and tested on the first set. For each training we limited the maximum number of partitions to four and the minimum number of utterances in each partition to 500, in order to reduce the risk of overtraining. The target *FA/all* was 7.7% for both the single-threshold system and the multi-threshold system. Performance was averaged on the two test sets and compared with the baseline single-threshold result, as shown Table 1.

	CA/all	FA/all
Single Threshold	29.0%	7.7%
Multi Threshold by Duration	29.7%	7.7%

Table 1. Listing recognition performance comparison between baseline single-threshold and auto-generated multi-threshold system based on utterance duration. All numbers were obtained by averaging metrics on the two test sets from a two-fold cross validation experiment.

	Very Short	Short	Medium	Long
Max Duration	1.66s	2.56s	3.60s	+inf.
Threshold	58	54	52	56

Table 2 Maximum duration of and confidence threshold for utterances in each of the four automatically generated data partitions based on utterance duration feature, from one of the two training data sets. Baseline single threshold is 56. Confidence values are integers between 0 and 100.

Note that this particular recognition task was very difficult as almost half of the listing utterances were effectively out of grammar, based on exhaustive manual validation. The 0.7% CA/all increase thus represented a substantial performance improvement over the single-threshold baseline.

Table 2 shows the duration boundaries and optimal threshold for each of the four automatically generated partitions, based on data from one of the training sets. Note that the baseline single threshold was *56*. Thus, the optimal threshold of *58* for "very short" utterances was found to be higher than the baseline whereas both "short" and "medium" utterances had lower-than-baseline threshold.

Figure 1 shows the individual *CA-FA* curve for each of the four partitions. It is particularly revealing to compare the confidence threshold movement from the baseline single threshold to the new threshold between "very short" and "medium" partitions. For "very short" utterances, raising threshold from *56* to *58* only slightly decreased *CA* rate but significantly decreased *FA* rate. This is represented by the relatively flat slope on the "very short" partition curve (solid line) between the two thresholds. On the other hand, for utterances in "medium" partition, lowering threshold from *56* to *52* significantly increased *CA* rate while only slightly increased *FA* rate. In this case, the slope between the two thresholds on the "medium" partition curve (dashed line) is fairly steep. Combining these two movements (plus a smaller threshold movement for "short" utterances) ultimately provided the overall performance improvement as in Table 1.

3.2. Discrete feature example

The second example uses a discrete feature in a phone number confirmation task. The system plays back to the caller a previously collected phone number and asks the caller to confirm whether or not the number is correct. The caller is supposed to answer "yes" or "no", but can also say the correct phone number directly or say "I don't know." In the baseline system, the confidence value associated with a recognition result is compared against a single threshold to determine whether the recognition result should be accepted. The objective is again to maximize *CA/all* and minimize *FA/all*.

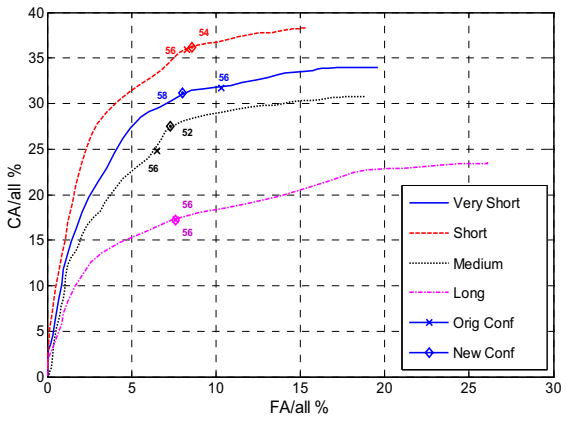


Figure 1. CA-FA curves of four individual data partitions from one of the two training data sets. "X" marks the operating point of baseline single confidence threshold; diamond marks the operating point of optimal threshold found for each partition by the multi- threshold algorithm.

Our data set for this task includes *3,050* caller utterances, a little over 20% of which are out of grammar, i.e. unexpected inputs such as speech from a side-conversation. Thus *CA/all* cannot exceed roughly 80%. Again, we randomly divided the data into two equal-sized data sets, performed a two-fold cross-validation, and averaged the multi-threshold performance on the two sets. The partitioning feature we used was the semantic interpretation class label. Each recognition result is uniquely classified as one of the following three classes: phone number string, "I don't know"-type utterance, and "yes/no" or other type of utterance.

In this experiment, the baseline single threshold was set to *40* (in a range between *0* and *100*), which had an *FA/all* of 7.4% on the entire data set. For the multi-threshold setting, however, we decided to target a significantly lower *FA/all* level at 3.0%. Table 3 shows the performance comparison between single threshold and multi-threshold systems.

	CA/all	FA/all
Single Threshold	76.0%	7.4%
Multi-Threshold by Semantic Class	76.3%	3.0%

Table 3 Phone number confirmation recognition performance comparison between baseline single threshold and auto-generated multi-threshold systems based on semantic class of recognition result. All numbers were obtained by averaging metrics on the two test sets from a two-fold cross validation experiment.

For this task, the multi-threshold system provided a significant reduction in *FA/all* while keeping a high *CA/all*. A close examination shows that instead of using the same confidence threshold of *40*, the multi-threshold system adopted a significantly higher threshold for phone number strings (*62*) and "I don't know" utterances (*53*), but a significantly lower threshold for "yes/no" utterances (*21*). Intuitively this suggests that the recognizer may have systematically overestimated confidence of phone number strings while underestimated confidence of "yes/no" output. This general trend happens to match very well what we have been experiencing with this particular recognition system.

4. Discussion and Conclusions

Experiment results above demonstrated the proposed multi-confidence thresholding algorithm can provide significant performance improvement to practical applications. The algorithm also has several desirable characteristics. First, it guarantees to find a multi-threshold setting that performs at least as well as, and often significantly better than, a single-threshold system on the same training data. This is always achievable since a single-threshold system is just a special case of multi-threshold where all partitions adopt the same threshold.

Secondly, the proposed algorithm can be applied to any speech recognition system that provides a confidence estimate along with recognition output, without the need to access the recognition engine internals. In fact, the algorithm is not limited to speech recognition, but can also be applied to improve the performance of any pattern recognition system with confidence estimation. How much performance improvement it can obtain depends on the particular recognition system and the specific task.

Thirdly, the proposed algorithm can use a variety of features that may have systematic impact on the confidence estimation in the underlying recognizer. Although we have only discussed the case of single feature, the algorithm can be extended to multiple features. For example, one may first apply the algorithm on one feature to find a set of partitions, and then use another feature on each of the found partitions. One should be cautioned that the greater complexity of feature partitions may lead to overtraining, particularly with insufficient training data.

Finally, although the proposed algorithm divides data into different partitions and adopts a different threshold for each partition, an alternative is to model the desired threshold with a functional form of the utterance feature, such as a regression formula (as reported in [5]). This may increase the modeling power of the algorithm but may also require a greater amount of training data.

5. References

[1] National Institute of Standard and Technology (NIST), SC-Lite – speech recognition scoring program. *http://www.nist.gov/speech/tools/index.html*.

[2] Gunawardana, A, Hon, H., and Jiang, L, "Word-Based Acoustic Confidence Measures for Large-Vocabulary Speech Recognition", In *Proceeding of ICSLP*98, 1998.

[3] Wessel, F, Schluter, Macherey, K. and Ney, H. "Confidence measures for large vocabulary continuous speech recognition", *In IEEE Trans. Speech and Audio Processing, vol. 9, no. 3, March, 2001.*

[4] Williams, D. A. *Knowing what you don't know: roles for confidence measures in automatic speech recognition.* Ph.D. Dissertation, University of Sheffield, May, 1999

[5] Gupta, S. and Soong, F. "Improved Utterance Rejection Using Length Dependent Thresholds", *In Proceedings of ICSLP98*, Sydney, *1998.*

[6] Chen, Z. and Ding, X. "Rejection algorithm for mis-segmented characters in multilingual document recognition", *In Proceedings of 7th Intl. Conf. on Doc. Analysis and Recognition*, vol. 2, *2003.*

[7] Russell, S. and Norvig, P. *Artificial Intelligence: a modern approach. 2nd Edition. Prentice Hall, 2002.*