Normalization of ASR Confidence Classifier Scores via Confidence Mapping

Kshitiz Kumar, Chaojun Liu, Yifan Gong

Microsoft Corporation, Redmond, WA

{kshitiz.kumar, Chaojunl, yifan.gong}@microsoft.com

Abstract

Speech recognition confidence classifier (CC) score quantitatively represents the correctness of decoded utterances in a [0,1] range. We associate an operating threshold with the classifier and accept recognitions with scores greater than the threshold. Speech developers may set their own threshold but often an acoustic model (AM) or CC update alters the correct-accept (CA) vs. false-accept (FA) profile, necessitating a threshold reselection. This is specifically a problem when, (a) threshold is hardcoded with a shipped hardware or software, (b) developers may not have expertise for threshold tuning, (c) tuning isn't cost-effective and may need to be done often. To our knowledge, our work is the first to present this practical and interesting problem of avoiding threshold reselection and proposes novel confidence-mapping-based techniques to improve or retain both CA and FA at previously set thresholds. We propose and evaluate, (a) histogram-based mapping, (b) polynomial-fitting, (c) tanh-fitting, based methods to map confidences associated with false-recognitions and discuss their issues and benefits. In our tests, all of the above mapping methods fix the mean regression in CA of 21% to a gain to 1-2%, with tanh-mapping providing the best CA and FA tradeoff in our tests.

Index Terms: Speech recognition, Confidence scores, Confidence predictors, Classifier, MLP

1. Introduction

Automatic speech recognition (ASR) has seen widespread use in recent years. Microsoft, Google, Apple, Nuance, University sites and speech developers have been pushing the reach of speech via developing practical speech applications. Although we strive for perfect recognition from ASR, we are far from human-like results as decoded utterances are invariably erroneous. ASR confidence classifiers play a critical role in the context of incorrect recognition results by quantifying reliability of the decoding. Confidence measure is especially important for applications where an ASR-enabled device is always in an active (continuously) listening mode in an application-constrained grammar. There potential recognitions from side-speech, background noise etc. can trigger unexpected system response, therefore, it's critical to prevent out-of-grammar (OOG) utterances from being recognized as in-grammar (IG) utterances.

The confidence classifier is trained to maximally discriminate between correct and incorrect recognitions. Confidence scores lie in a [0,1] range, we desire higher scores for correct recognitions, and lower for, (a) incorrect recognitions from in-grammar and, (b) any recognition from out-of-grammar utterances. The classifiers are trained from a specified set of acoustic model (AM), grammar and speech data, this establishes classifier profile in terms of CA and FA at different thresholds. We associate an operating threshold with the classifier and accept recognitions with scores greater than the threshold, upon which we evaluate correct-accept (CA) and false-accept (FA) measures.

We refer [1] for an introduction to our confidence classifier framework. There we also discussed our predictors confidence classifier approaches. We refer [2, 3, 4, 5, 6, 7, 8] for a survey of other confidence techniques and specifically [9, 4, 10, 11, 12] for predictors and the classifiers used.

In this work we present a new problem that is becoming increasingly relevant in speech applications. As per current design, confidence thresholds are inherently tied to the set of acoustic models and confidence classifiers (CC). Any change in the parameters of this set may improve or retain overall CA vs. FA tradeoff, but may change CA or FA at previous operating thresholds, necessitating retuning of thresholds for desired behavior. This is clearly critical in many practical situations where speech developers (3^{rd}-party) and our partner teams (1^{st}-party) may not have adequate expertise and data to retune thresholds. To our knowledge, our work is the first to present this practical and interesting problem of avoiding threshold reselection and proposes novel confidence-mapping-based techniques to simultaneously improve or retain both CA and FA at previously set thresholds. Rest of this work is organized in the following. We explicitly discuss the relevance of confidence normalization in Sec. 2. We present confidence normalization techniques in Secs. 3, 4, and 5 and present related experiments and results in Sec. 6. Sec. 7 concludes this study.

2. Relevance of Confidence Normalization

In this Section, we present a very practical and interesting problem of confidence normalization and discuss its relevance. We specifically highlight our problem with respect to the chart in Fig. 1. There we plot confidence profile in CA vs. FA for an *OldModel* and *NewModel* where compared to OldModel, NewModel provides better CA for every FA. There $CA = \frac{\#All\ Corrects\ beyond\ a\ threshold}{\#All\ Corrects}$, and $FA = \frac{\#All\ Incorrects\ beyond\ a\ threshold}{\#All\ Incorrects}$, where $\#$ indicates count. The points on the curve indicate different operating thresholds, an overall improvement in CA vs. FA expectedly hides operating thresholds, and doesn't necessarily guarantee that CA for NewModel is better than that for OldModel at every operating threshold. In Fig. 1 we draw a line to join the points on the 2 curves corresponding to threshold 65. Considering 65 as set threshold, NewModel provides a lower FA than OldModel but also regress CA from 81% to 48%. Thus our ASR update will lead to severe CA regression for our 1^{st}-party or 3^{rd}-party speech developers. This is undesirable and can hurt end-user experience. We can also have alternate situations where a model update may improve CA but regress FA.

The issue of confidence normalization additionally arises in following situations, (a) threshold is hardcoded with a shipped software *e.g.* games, applications, (b) developers may not have

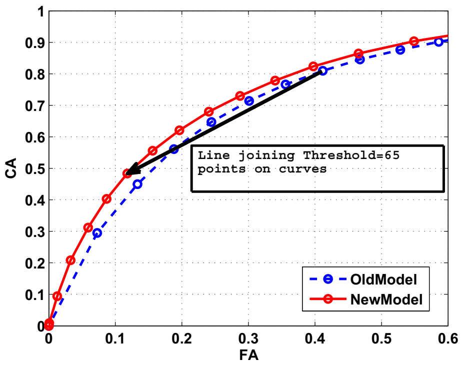

Figure 1: *Confidence profile in CA vs. FA for models in* OldModel *and* NewModel. *The dots there indicate CA and FA values at different thresholds, we draw a line to join dots for* $threshold = 65$.

expertise or data for threshold tuning, (c) it is preferable to untie dependency of operating thresholds on AM or CC, then AM update can occur multiple times without having to update thresholds, *e.g.* this is the case for our *Xbox* partners, where we may update AM or CC, without creating a dependency on our partners to retune thresholds, (d) all 3^{rd}-party speech applications may once set their own operating thresholds, thus an AM update leading to threshold tuning can potentially have a huge downstream cost for the developers, (e) tuning isn't resource or cost-effective and may need to be done often for every little speech application.

Thus, we need to ensure that the new CA vs. FA profile for an AM or CC update provides:

1. better CA with same FA, *i.e.* every corresponding dot for NewModel in Fig. 1 be strictly above the dot for OldModel, or
2. better FA with same CA, *i.e.* every dot for NewModel be strictly on the left of the dot for OldModel, or
3. better CA and better FA, *i.e.* every dot for NewModel be on the upper-left of the dot for OldModel.

In this work we choose to propose our solution to address either 1^{st} or 2^{nd} situation in above, as we believe these are the typical cases and required by our partners as well. For the rest of the paper, we only talk about the 1^{st} situation, *i.e.* mapping confidences to achieve same FA at every threshold such that the new curve gets better CA. Our techniques can exactly be applied to 2^{nd} situation as well.

We will apply this work for every potential Microsoft (MS) AM or CC update that may alter the CA vs. FA profile. This will improve confidence performance without requiring threshold tuning by speech developers for unchanged application grammars. We find the confidence profile to be robust to minor grammar changes but sometimes developers may do major grammar updates that can alter the confidences. To address this, MS can transfer the normalization tool to developers. Our normalization work requires the presence of OldModel and NewModel, along with calibration data. Developers can either collect data or use simulation or text-to-speech (TTS) techniques to obtain calibration data according to updated grammars, then the MS tool can easily provide them the confidence normalization parameters. This work can be applied to non-MS products as well.

3. Histogram-based Mapping

In this Sec. we study our problem presented in Sec. 2 in a histogram equalization (HEQ) framework. Histogram equalization [13] is a popular image processing technique, it learns a mapping on input image pixels to result in an output image with increased contrast. Histogram equalization works by spreading the pixels to uniformly cover the image intensity values. HEQ is also applied in speech recognition [14] to equalize speech features. For our work we use the fundamentals of histogram equalization and apply that for confidence mapping. While usually the target is a uniform distribution for HEQ-based contrast enhancement, we choose our target as distribution of confidences for false-recognitions from OldModel, with corresponding confidences from NewModel being inputs to equalization. We further demonstrate the problem of confidence normalization with respect to the confidences histograms for *OldModel* and *NewModel* in Fig. 2.

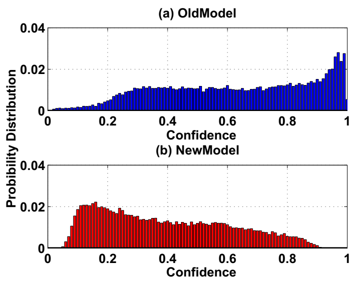

Figure 2: *Histogram of confidences associated with false-recognitions from* OldModel *and* NewModel.

Clearly, the different FA profiles in Fig. 1 translates to different histograms in Fig. 2, where the confidences for NewModel are noticeably lower than that for OldModel, there the respective mean confidences are 0.30 for NewModel and 0.63 for OldModel. The goal of the histogram-based mapping is intake confidences for false-recognitions from NewModel and output confidences whose distribution matches to those from OldModel. We describe our histogram-based mapping algorithm in below:

1. In $P_O(t)$ obtain probability mass function (PMF) for confidences associated with false-recognitions for OldModel $\forall\, t$ in $T = [0 \ldots 1]$, at specified steps of δ. Similarly $P_N(t)$ holds PMF for NewModel.
2. Corresponding to PMFs in above, obtain cumulative mass functions in $C_O(t)$ and $C_N(t)$ for respectively OldModel and NewModel at all thresholds t.
3. Obtain t_{map} to yield $C_O(t_{map}) = C_N(t_{in}), \forall\, t_{in}$ in T.

Thus we learn a table-lookup based mapping that yields a corresponding t_{map} for every input t_{in} in $[0 \ldots 1]$. Here the step size δ limits the mapping resolution. Based on above mapping, we apply histogram mapping on test data by binning input confidences into one of the bins in $t_{in} \in T$ and selecting corresponding t_{map} as normalized confidence. We compare our mappings in Fig. 4 and discuss results in Sec. 6.

4. Polynomial-based Mapping

Histogram-mapping in Sec. 3 addressed the problem of confidence mapping in a non-parametric approach by storing a table of input and output values. Here, we seek a parametric solution by explicitly learning a mapping function between the input and output confidences. For unconstrained memory, histogram-mapping can achieve arbitrarily high degree of mapping with adequate δ but in practise we are limited in terms of memory and computation. Further there is a tradeoff with overfitting training data, that may not generalize to test data. In this context it is relevant to seek parametric solutions that can be described with fewer parameters and may not overfit training data. We choose a polynomial as our mapping function and evaluate its parameters over the same set of data used for histogram mapping in Sec. 3. We describe the polynomial-based mapping in below:

1. In $F_O(t)$ obtain FAs for OldModel $\forall\ t$ in $T = [0 \ldots 1]$, at specified steps of δ_T. Similarly $F_N(t)$ holds FAs for NewModel.
2. Sample to obtain $T_O(f)$ and $T_N(f)$ for $\forall\ f$ in $F = [0 \ldots 1]$, at specified steps of δ_F. Fig. 3 shows this intermediate step.
3. Via least squares regression, learn a polynomial to yield $T_O(f) = \sum_{i=0}^{D} a_i * T_N(f)^i\ \forall\ f$ in F, with parameters in a_i.

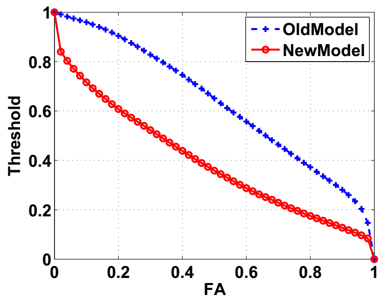

Figure 3: *Operating threshold at different FAs for* OldModel *and* NewModel.

During evaluation, we simply apply the learned polynomial mapping over input confidences to yield output confidences, there we chose $\delta_T = \delta_F = 0.01$. Compared to the table that needs to be stored in histogram-mapping, the memory requirement for polynomial mapping is minimal, we only store $D + 1$ float values, where in practise D may be in 2-4 range.

An issue with polynomial-fitting is that it may not necessarily yield a monotonic mapping, this is unexpected in speech application since greater input confidences must not yield a lower output confidence. In practise we see this non-monotonicity to be an issue for extreme input confidences only, *i.e.* confidences lower than 0.025 or greater than 0.975, even there we see output confidences to be within 0.005. To solve non-monotonicity issue, we may add monotonicity as a constraint in polynomial-fitting.

5. Tanh-based Mapping

In this Sec. we build on the parametric mapping presented in Sec. 4 and present an alternate mapping that uses tanh function. Our MLP confidence classifiers use tanh nonlinearity, there the output confidence c for penultimate hidden layers with J nodes with outputs h_j and corresponding MLP weights w_j^o and bias b^o is:

$$c = \frac{1 + \tanh(b^o + \sum_{j=1}^{J} w_j^o h_j)}{2} \quad (1)$$

Above formulation guides to using tanh as an alternate mapping that also addresses the monotonicity issue with polynomial-fitting. Here, the confidence mapping can be achieved via the scale and bias parameters in the argument of tanh. We present our approach in:

1. Let c_O and c_N represent corresponding confidences for false-recognitions from OldModel and NewModel.
2. Via least squares regression, learn bias (s_0) and scale parameter (s_1) to obtain: $atanh(c_O) = s_0 + s_1 \cdot atanh(c_N)$, where $atahn$ is inverse of tanh.

6. Experiments and Results

We provide the mappings learned for the mapping functions presented in Secs. 3, 4, and 5 in Fig. 4. These mapping were obtained on a calibration data with 200k words by normalizing the confidences associated with false-recognitions for *NewModel* to reflect those for *OldModel*. There histogram-normalization reflects the ground truth, there we chose step-size δ as 0.01. We see that both polynomial and tanh fitting closely follow the histogram-fitting.

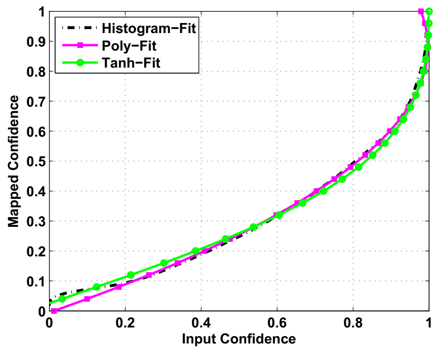

Figure 4: *Mapping to normalize confidences.*

Next we apply the mapping function on 4 blind tests *Test1-4* and report results in Figs. 5 and 6. These are Microsoft internal tests for US-English voicesearch and each consist of above 50k words. We see that all of the mapping techniques fix the regression in CA with *NewModel* over *OldModel* at all thresholds.

We report mean difference in FA (*i.e. without modulus operation*) across operating thresholds in terms of ($FA_{Mapped} - FA_{OldModel}$) metrics across 4 tests and their averages in Table 1. Here the FA mapping target was to be as close as possible to $FA_{OldModel}$, there all of the mapping functions on average provide a desirable lower FA, with Tanh-Fit being the best. As mapping techniques become stronger, the mean difference should converge to zero.

In contrary to Table. 1, we evaluate mean CA difference across thresholds in terms of ($CA_{Mapped} - CA_{OldModel}$) metric. These values are respectively -0.214, 0.016, 0.019, and 0.022 for *NewModel*, Hist-Fit, Poly-Fit and Tanh-Fit. Here too we see that the Tanh-Fit provides better overall CA gain among all mapping methods.

Corresponding to above mappings we provide detailed results with respect to FA metrics across calibration and test in

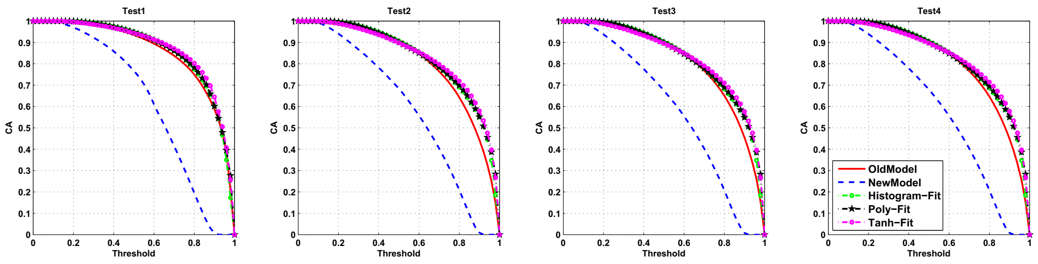

Figure 5: *CA vs. Threshold results for the mapping techniques developed.*

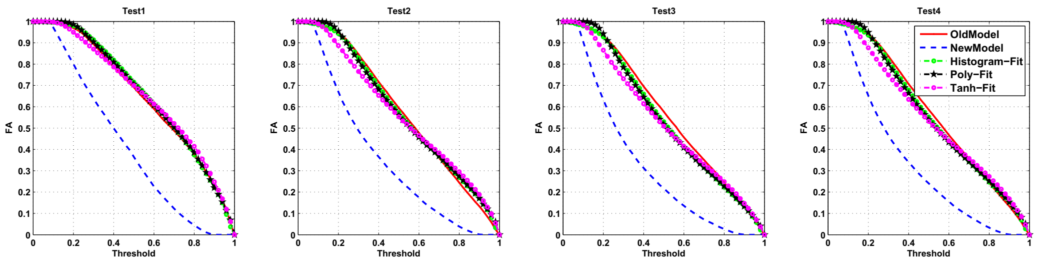

Figure 6: *FA vs. Threshold results for the mapping techniques developed.*

Table 1: Mean difference in FAs *i.e.*$(FA_{Mapped} - FA_{OldModel})$, lower value is better.

Method	Test1	Test2	Test3	Test4	Average
NewModel	-0.2331	-0.2251	-0.2411	-0.2329	-0.2330
Hist-Fit	0.0109	-0.0012	-0.0211	-0.0109	-0.0056
Poly-Fit	0.0104	-0.0014	-0.0210	-0.0109	-0.0057
Tanh-Fit	0.0082	-0.0116	-0.0336	-0.0223	-0.0148

Table 2: Mean difference in FAs *i.e.* $(FA_{Mapped} - FA_{OldModel})$ across tasks, lower value is better.

Metric	Hist-Fit	Poly-Fit	Tanh-Fit
on calibration (w/ mod)	0.0105	0.0120	0.0218
on test data (w/ mod)	0.0170	0.0220	0.0334
on test data (w/o mod)	-0.0056	-0.0057	-0.0148

Table 2. Since the mapping objective on calibration data is to map confidences to a target value, there we report FA difference in absolute sense, *i.e. with modulus operation*; expectedly Hist-Fit has the lowest FA difference (0.0105) . For test data, the mean without any modulus operation is more meaningful since we do not want to penalize datapoints where mapped FA is lower than that for *OldModel*, then we see Tanh-Fit to be the best. Overall all of the 3 mapping techniques solve the problem of normalizing confidences with AM or CC update.

6.1. Imposing a Meaning on Confidences

In this Sec. we apply our mapping work to impose a meaning to the confidence thresholds. As discussed, these thresholds are usually derived from CA vs. FA profile to satisfy an operating criterion, these thresholds by themselves have little practical connotation other than implying greater confidence in recognitions with higher thresholds. In Fig. 7 we apply histogram-fit to confidences of NewModel on Test1 data with mapping target as $CA = (1 - Threshold)$ at all operating thresholds. Thus an application setting a threshold of 0.10 will achieve a CA of 0.90, and similarly for all other thresholds. We can also do a mapping on confidences to imply a meaning on thresholds with respect to FA by setting mapping target as $FA = (1 - Threshold)$. Either of these approaches add an intuitive meaning to otherwise less informative threshold values.

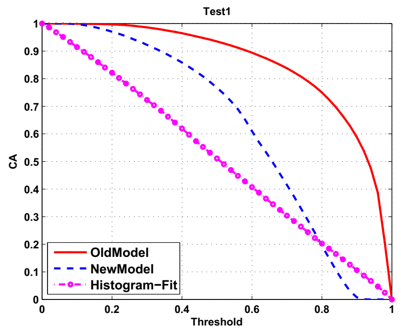

Figure 7: *Imposing meaning to confidences by mapping confidences associated with correct recognitions.*

7. Conclusions

We introduced a new problem of confidence normalization in this work. We presented various scenarios where an AM, CC or grammar update may alter CA or FA associated with thresholds, potentially regressing confidence performance at set thresholds. This problem is of immense practical importance for our 1^{st} and 3^{rd}-party speech developers. We studied the problem via mapping confidences associated with either FA or CA and presented mapping techniques in histogram-fitting, polynomial-fitting and tanh-fitting. We compared the techniques with respect to mean difference in FAs across thresholds. In our tests, all of the mapping methods fixed the mean regression of 21% in CA at set thresholds, among which histogram mapping was expectedly the best on calibration data but tanh-fitting was better on test data. We also applied our mapping techniques to impose a meaning on the confidence thresholds.

8. References

[1] P.-S. Huang, K. Kumar, C. Liu, Y. Gong, and L. Deng, "Predicting speech recognition confidence using deep learning with word identity and score features," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 7413–7417.

[2] H. Jiang, "Confidence measures for speech recognition: A survey," *Speech Communication*, vol. 45, no. 4, pp. 455 – 470, 2005.

[3] F. Wessel, K. Macherey, and H. Ney, "A comparison of word graph and n-best list based confidence measures," in *Proc. of EuroSpeech*, 1999, pp. 315–318.

[4] C. White, J. Droppo, A. Acero, and J. Odell, "Maximum entropy confidence estimation for speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2007, vol. 4, pp. 809–812.

[5] J. Blatz, E. Fitzgerald, G. Foster, S. Gandrabur, C. Goutte, A. Kulesza, A. Sanchis, and N. Ueffing, "Confidence estimation for machine translation," in *M. Rollins (ED.), Mental Imagery*. 2004, Yale University Press.

[6] R.C. Rose, B.-H. Juang, and C.H. Lee, "A training procedure for verifying string hypotheses in continuous speech recognition," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, May 1995, vol. 1, pp. 281–284.

[7] L. Mathan and Laurent Miclet, "Rejection of extraneous input in speech recognition applications, using multi-layer perceptrons and the trace of hmms," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, Apr 1991, pp. 93–96 vol.1.

[8] R.A. Sukkar and Chin-Hui Lee, "Vocabulary independent discriminative utterance verification for nonkeyword rejection in subword based speech recognition," *Speech and Audio Processing, IEEE Transactions on*, vol. 4, no. 6, pp. 420–429, Nov 1996.

[9] T. Kemp and T. Schaaf, "Estimating confidence using word lattices," in *Proc. of EuroSpeech*, 1997, pp. 827–830.

[10] F. Wessel, R. Schlter, K. Macherey, and H. Ney, "Confidence measures for large vocabulary continuous speech recognition," *IEEE Trans. on Speech and Audio Processing*, pp. 288–298, 2001.

[11] L. Chase, "Word and acoustic confidence annotation for large vocabulary speech recognition," in *in Proc. European Conference on Speech Communication Technology*, pp. 815–818.

[12] M. Weintraub, F. Beaufays, Z. Rivlin, Y. Konig, and A. Stolcke, "Neural-network based measures of confidence for word recognition," in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, 1997, pp. 887–890.

[13] R. C. Gonzlez, *Digital Image Processing*, Prentice Hall, 2007.

[14] A. De la Torre, A.M. Peinado, J.C. Segura, J.L. Perez-Cordoba, M.C. Benitez, and A.J. Rubio, "Histogram equalization of speech representation for robust speech recognition," *Speech and Audio Processing, IEEE Transactions on*, vol. 13, no. 3, pp. 355–366, May 2005.