Confidence-Features and Confidence-Scores for ASR applications in Arbitration and DNN Speaker Adaptation

Kshitiz Kumar, Ziad Al Bawab, Yong Zhao, Chaojun Liu, Yifan Gong

Microsoft Corporation, Redmond, WA

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

Abstract

Speech recognition confidence-scores quantitatively represent the correctness of decoded utterances in a [0,1] range. Confidences are primarily used to accept recognitions with scores greater than a threshold and thus contain recognitions from background noise or out-of-grammar (OOG) speech. Confidence scores have also been used in other speech applications, (a) Rover where we do multi-system combination, (b) Arbitration where we pick one among multiple simultaneous recognitions, (c) selecting high quality data for unsupervised model training etc. Confidence-scores are computed from a rich set of confidence-features in the speech recognition engine. While many speech applications consume confidence scores, we haven't seen adequate focus on directly consuming confidence-features in applications. In this work we build a thesis that additionally consuming confidence-features can provide huge gains across confidence-related tasks and demonstrate that with respect to application to Arbitration, where we present 30% relative reduction in arbitration metric. We also demonstrate an application of confidence-score in deep-neural-network (DNN) speaker adaptation metric, where we can double relative reduction in word-error-rate (WER) for DNN speaker adaptation on limited data.

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

1. Introduction

Automatic speech recognition (ASR) has seen a huge wave of deployment and usage across devices and services in recent years. Confidence-scores are integral component of ASR, these scores quantify the correctness of recognition results. Historically confidences were used for ASR-enabled devices that are 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. Confidence-scores are typically evaluated for a word as well as for an utterance.

Recently confidence-scores have also been used in other ASR applications in (a) system combination using ROVER, (b) selecting one of client or service recognition in Arbitration, (c) selecting high-quality data for unsupervised model training, (d) Key-word spotting, (e) confidence-normalization etc. While many of the downstream speech applications consume confidence-scores we have seen limited attempts on additionally consuming confidence-features. In this work we present our individual confidence-features, and, discuss the diverse information that they capture. We specifically demonstrate the richness of these features for Arbitration application where we present significant gains in arbitration metric. We also present a DNN speaker adaptation application where we include confidence-scores in adaptation framework.

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 [**?**] for an introduction to our confidence classifier framework. There we also discussed our predictors confidence classifier approaches. We refer [**?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**] for a survey of other confidence techniques and specifically [**?**, **?**, **?**, **?**, **?**] for predictors and the classifiers used.

Rest of this work is organized in the following. We explicitly discuss the relevance of confidence normalization in Sec. **??**. We present confidence normalization techniques in Secs. **??**, **??**, and **??** and present related experiments and results in Sec. **??**. Sec. 7 concludes this study.

2. Background on Confidence-Features and Confidence-Scores

We discussed the significance of confidence-score in Section 1 where we mentioned the job of the confidence-classifier is to make an inference on the correctness of recognition events. This is thus a binary classification problem [**?**] with the 2-classes in (1) correct recognitions, (2) all incorrect recognitions that includes misrecognitions over IG utterances as well any recognition from OOG utterances. The classifier is trained from a rich set of confidence-features that we obtain from speech decoding. Some of the prominent features are:

1. acoustic-model features - we aggregate per-frame acoustic score over a word or an utterance. We also compute scores from acoustic arc transitions. These scores are typically normalized with respect to duration and
2. language-model features - we compute language fanout, perplexity
3. noise and silence-model features - we compute features from noise and silence models
4. 2nd order features - we can compute weighted average of above features with respect to words in a phrase

5. duration features - we can compute number of words in a phrase, word-duration etc.

Some other features may include count of senones from decoding. Above features are appropriately normalized to be robust to speech with different duration and intensity. Though confidence-scores have been used in a number of speech applications, we have seen limited focus on directly consuming confidence-features in the applications. The count of these features is typically 15-20 for a word or for an utterance, so additional memory required to store these features is minimal. Most of the current ASR engine just output recognition text along with confidence. These confidence-features are already computed for the purpose of confidence-score so there is almost no additional work required for extracting confidence-features. Furthermore, these confidence-features also need to be transferred alongwith ASR result to the consumer to these features. Considering a typical speech segment of 4 seconds encoded at 8 kB/second for a total of 32kb, confidence-features just add 60 Bytes to the footprint, thus less than 0.2% to overall footprint.

We set up a large pool of training data, grammar, and obtain positive tokens from successfully recognized utterances and negatives from wrong recognitions. Based on above set of positive and negative features, we build a multi-layer perceptron based classifier (MLP) [**?**, **?**]. This uses a single hidden layer with specified set of nodes and is trained with mean-squared error (MSE) criterion. We can build a classifier for either word or sentence. For word classifier, predictors are aggregated and averaged across a word boundary, similarly predictors are aggregated over a sentence for corresponding classifier. Our confidence-features consist of 16 individual features. We refer to [**?**] for additional details on our confidence classifier architecture.

3. Rich Confidence-Features for Arbitration

Arbitration is an application where we expect to select the best among one of the multiple simultaneous ASR results. We explain our arbitration framework for Microsoft Cortana experience on smart devices in Fig. 1, there we select the best result between Client and Service side results. Client engine is designed to work with traditional client scenarios like *call, digit dialing, text, open applications etc.* and service engine works for rest of the speech scenarios including *dication, voice-search, weather etc.*. By design the client and service cater to distinct speech scenarios and contain language-model and acoustic-model optimized for those tasks. Though we have distinct acoustic models that cater to client and service speech scenarios, they both work together in a unified way and indistinct for the user as we obviously don't expect user to provide us inputs on whether his scenario is for client or for service. Arbitration is the key speech component that provides this unified experience for user by selecting the best among the client and service results. So although the client and service engines are designed for their own respective scenarios, they listen to speech from potentially all scenarios and produce respective recognition results, these results are consumed by arbitration where it selects the best between the client and service results.

Arbitration sends the results back to client where a decision unit at client will typically provide the arbitrated result to the user on their smart devices. There can be a few scenarios where the decision unit can simply choose the client recognition viz. (a) If the client confidence-scores are higher than a present threshold - typically the process of getting the arbitrated ASR result back from service may incur some latency so client can simply choose client ASR result if it's very confidence about it, (b) when we have no connection to service - in these cases user can at least make use of client side speech applications from Microsoft or other 3^{rd}-party applications. This framework also allows us to cater to scenarios where a user says: "text Alex, I will be late". There client engine has access to contacts and can recognize the person name "Alex" and produce a result like "text Alex ...", then "..." will be filled in by recognition from service.

As noted in Fig. 1, speech from user is sent to both client and service, where they both produce ASR results under the constraint of their respective engine, AM and LM. Client AM contains garbage paths that greatly help absorb utterances are intended for service only and thus are out-of-grammar (OOG) for client.

4. Arbitration Classifier and Baseline Features

Brief description of current arbitration framework, and current useful features

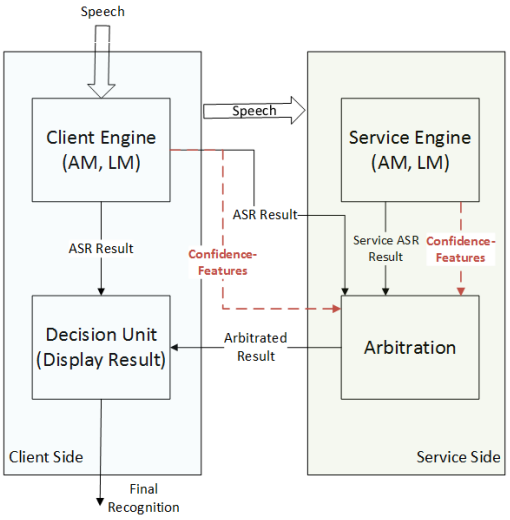

Figure 1: *Arbitration for client and service ASR resuls.*

4.1. Incorporating Confidence-Features in Arbitration

We described our arbitration framework and the baseline features in 4, there we mentioned a number of features that we found very relevant for arbitration. In this section, we build a thesis on extracting the rich set of confidence-features and propagating them for use in arbitration. We first note that although our arbitration baseline features provide very useful information, the confidence-features provide additional information in terms of noise, silence, acoustic and language-model scores, all this is expected to useful for arbitration. We also note that although confidence-scores from both client and service is used by arbitration, and expected to provide a good gist of confidence-features, we can still benefit a lot with directly using confidence-features by, (a) using much more gradual information in terms of 15-20 confidence-features vs. using a single confidence-score, (b) confidence-scores are designed to

optimize the performance confidence-classifier which is clearly different from arbitration, so retraining with the confidence-features can help, (c) arbitration and confidence-classifier may be trained over different datasets, so the information encapsulated by confidence-score may not generalize to dataset relevant for arbitration, (d) confidence-scores are language-specific as they may have been individually trained across a set of *AM, LM, locals, dataset*, in contract we have noted that the various inherent normalizations in confidence-features make them robust across locals, so using confidence-features can allow us to build an arbitration classifier from one local that can provide good performance for other unseen locals, this can be specially useful when we may need to bootstrap arbitration from a local under limited data scenarios, (e) using confidence-score in arbitration creates a dependency for arbitration on confidences, any update to confidence-classifiers potentially requires retraining arbitration, we can completely alleviate this issue if we directly consume confidence-features in arbitration while not using confidence-score feature, this decouples arbitration and confidence-classifiers, letting us independently update confidence-classifier without impacting arbitration.

We demonstrate our approach that uses the rich confidence-features for arbitration in Fig. 1. We build the infrastructure required to extract confidence-features from both client and service engine, and send them both to arbitration. As expected we require little work for additionally communicating service confidence-features to arbitration. For client, we need to additionally send about 30 bytes-per-second of data, this is only about 0.2% relative increase to our payload from client to service. Depending on application and need, arbitration can be retrained and deployed with confidence-features from (a) just client, (b) just service, (c) both client and service.

4.2. Arbitration Experiments and Results

We present and analyze results from using confidence-features in arbitration as discussed in Sec. 3. Our arbitration was trained from over 35k speech utterances. Testing was done on over 25k utterances. We decoded these utterances against both client and service engines and obtained corresponding recognition results. We had ground-truth transcriptions for these utterances and created their classification targets in terms of client or service based on the WER criterion. We obtained the arbitration baseline-features that included confidence-score, and additionally obtained confidence-features from both client and service. Note that ideally clients will have distinct personalized grammars with their own contact names, application names etc. For our purposes we simulated client grammar with each over 250 names in contacts and used these grammars in decoding. We followed the existing framework for training arbitration classifier where we additionally used confidence-features.

Next, we demonstrate the value in client confidence-features in Fig. 2, there we plot features distribution for a few features for arbitration task. There "correct" refers to cases where client wins, and "InCorrect" refers to service wins, "Confidence-Score" indicates usual client confidence-score. We visually see that some of the features much better separate the 2 classes than confidence-score.

Next we provide receiver-operating-curve (ROC) for baseline and with including client confidence-features in Fig. 3, where we note a strongly better ROC curve through the range of curve. In this arbitration task, "False Positive" (FP) indicates incorrect wins from client and "True Positive" (TP) indicates correct wins from service. Specifically at FP of 0.1, we can

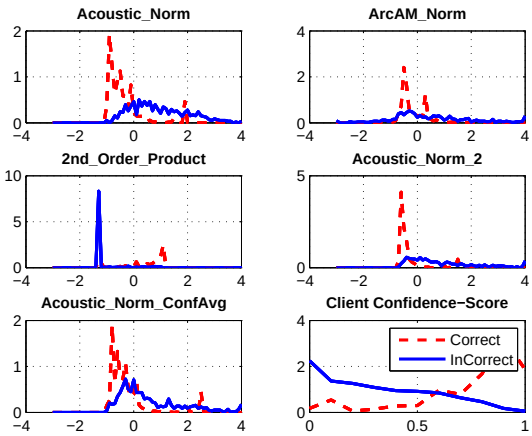

Figure 2: *Mapping to normalize confidences.*

improve TP from 0.81 to 0.86, for a 26% relative reduction in (1-TP). We note correct area-under-the-curve (AUC) metrics in Table 1, where including client confidence-features improved AUC from 0.927 for baseline to 0.946, additionally including server confidence-features improved AUC to 0.95 for a 31.5% relative reduction in (1-AUC) metric.

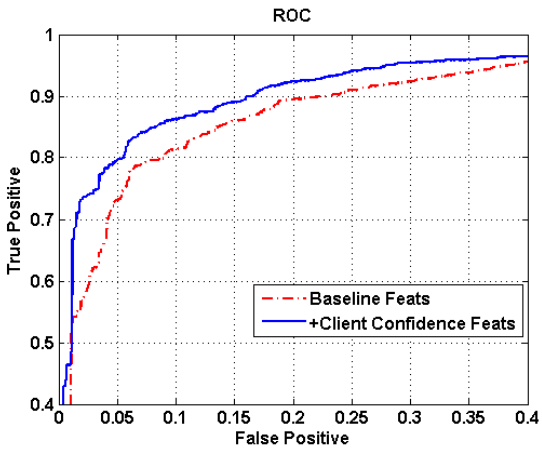

Figure 3: *Mapping to normalize confidences.*

Our arbitration classifier ranks all of it's features in the order of importance. As expected we see that the confidence-features appear prominently in the top features, with 7 of the top-10 overall features being confidence-features. This further demonstrates that the proposed features outperform the baseline features.

5. Confidence-Scores in DNN Speaker Adaptation Framework

We have noted that speaker adaptation is an area where we have significant avenue for further improvements. In this section, we propose a novel application of confidence-score for DNN speaker adaptation work for limited and large adaptation data scenarios. We have noted that confidence-scores imply the correctness of recognition results. In the context of

Table 1: Area-under-the-curve (AUC) for ROC chart

Method	AUC	relative reduction in (1-AUC) [%]
Baseline Features	0.927	-
+ Client Confidence-Features	0.946	26.0
+ Service Confidence-Features	0.950	31.5

Table 2: WER for Supervised adaptation. Baseline WER is 19.9%.

Nutts	Best approach		+Include Confidence	
	WER	WERR	WER	WERR
20	19.6	1.5	18.9	5.0
50	17.6	11.6	17.1	14.1
100	16.7	16.1	16.4	17.6

speaker-adaptation confidences also indicate a degree of match between the speaker-dependent data and speaker-independent model. Thus we can leverage confidence-score in the DNN speaker adaptation optimization by disproportionately weighting data across confidence-scores. Correspondingly we created a new recipe where we first bucket all of adaptation data into 3 buckets for low, medium and high confidence-scores. Our goal is to alter the optimization metric by including confidence-scores. Corresponding to the 3 confidence-categories we can weight the data samples from those categories according to specified values for those categories.

We know that confidence can indicate a great deal about the quality of utterances that we use in adaptation but none of the current adaptation recipes include confidence, specifically: For correct hypothesis For incorrect hypothesis, (a) low confidence data is a poor match to model and may benefit with higher weight on the data, (b) low confidence results are likely to be incorrect, so we should deemphasize these utterances, (c) high confidence data is already a good match to model, so there is less to learn from this data, (d) high confidence results are likely to be correct, so we should emphasize these utterances.

For 50 utts we can improve WERR from 11.6% to 14.1% for supervised adaptation.

Approach. New confidence recipe Split an utterance alignment into individual words Group the words bases on their confidences in low, medium and high confidence buckets We can disproportionately duplicate low, medium and high confidence words Testing remains unchanged, where we test on the entire utterance without splitting them into word-level Applied this recipe to training and testing on 6 speaker adaptation data on SMD task Based on experiments selected a recipe that simply duplicates the low and medium confidence buckets while retaining the high confidence bucket. Noting results for unsupervised and supervised adaptation in below

5.1. DNN adaptation experiment

Training vs test Dataset Server task Large LM Any difference of above 1% relative is significant.

6. Discussion

We demonstrated novel applications of confidence-features and confidence-scores in this work, where we presented strong gains for arbitration and DNN speaker adaptation. We are also investigation applications of these confidence-features to speech applications in ROVER for system combination and for model recommendation for identical LM we can recommend one of the many AMs that may work best in particular scenarios. We are seeing promising gains in above applications.

7. Conclusions

Speech recognition confidence-scores quantitatively represent the correctness of decoded utterances in a [0,1] range. Confidences are primarily used to accept recognitions with scores greater than a threshold and thus contain recognitions from background noise or out-of-grammar (OOG) speech. Confidence scores have also been used in other speech applications, (a) Rover where we do multi-system combination, (b) Arbitration where we pick one among multiple simultaneous recognitions, (c) selecting high quality data for unsupervised model training etc. Confidence-scores are computed from a rich set of confidence-features in the speech recognition engine. While many speech applications consume confidence scores, we haven't seen adequate focus on directly consuming confidence-features in applications. In this work we build a thesis that additionally consuming confidence-features can provide huge gains across confidence-related tasks and demonstrate that with respect to application to Arbitration, where we present 30% relative reduction in arbitration metric. We also demonstrate an application of confidence-score in deep-neural-network (DNN) speaker adaptation metric, where we can double relative reduction in word-error-rate (WER) for DNN speaker adaptation on limited data.