INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Hybrid Arbitration using raw ASR string and NLU information – taking the best of both embedded world and cloud world

Min Tang

Nuance Communications, Bellevue, Washington, USA
Min.Tang@nuance.com

Abstract

Hybrid arbitration is a process where we select the best Automatic Speech Recognition (ASR) and Natural Language Understanding (NLU) result from embedded/client and cloud-based system outputs. It is a common approach that a lot of real world applications use to unify knowledge sources that are not available to client and cloud at the same time. In the past, people primarily relied on ASR confidence features and some application specific heuristics in the arbitration process. However, confidence features are unable to capture subtle context specific differences. In this paper, besides confidence, we also use raw ASR strings and NLU results in the hybrid arbitration process. We model the arbitration process as two steps – first, decide whether to wait for a slower system, and second, pick the best result. We compared multiple machine learning approaches and it turns out the Deep Neural Network (DNN) based classifier, using word embeddings to process ASR strings and NLU embeddings to process NLU information, can deliver the best performance. We conducted experiments on two production system setups, using field data from real users. Compared with traditional confidence score based approach, we obtain about 30% relative word error reduction and 30% relative sentence error rate reduction.

Index Terms: System Arbitration, DNN, Classification

1. Introduction

In recent years, with the advances of deep learning technologies in speech recognition [1, 2, 3] and the availability of affordable broadband internet to mobile devices, more and more commercial voice interface systems, especially automotive voice assistant systems, are moving towards the hybrid ASR architecture, which combines both embedded and cloud systems based on latency tolerance, cost, security, computation power, and network capacity, enabling end-user's queries to be simultaneously processed in embedded and cloud systems for optimal results.

Although a cloud ASR/NLU system has the advantage of powerful and flexible computing resources, a lot of times it is missing the access to end-user's personal data, because users of speech-driven solutions have become increasingly concerned about sharing their personal on-device data, such as contacts or music, with on-line services, even though personal data is very important for an ASR/NLU system to deliver highly accurate results. On the other hand, it is very common for an embedded ASR/NLU system to access on-device user data and incorporate them in models. There are multiple ways to combine embedded and cloud systems while preserving privacy, e.g., (a) [4] proposed a framework for secure speech recognition by using privacy preserving hidden Markov model and Gaussian mixtures computations, (b) [5] performed first pass decoding on cloud, and then fine tuned the result by doing second pass decoding on device using dynamic language models built from personal data, (c) lattice-based system combination approaches to combine embedded ASR result and cloud ASR result, [6, 7, 8], and (d) system arbitration approaches to combine embedded ASR/NLU result and cloud ASR/NLU result. In this paper, we are focusing on (d), since it is the most straight-forward and easiest way to take the best of both embedded world and cloud world. Fig.1 shows the hybrid ASR architecture in a typical automotive voice assistant system.

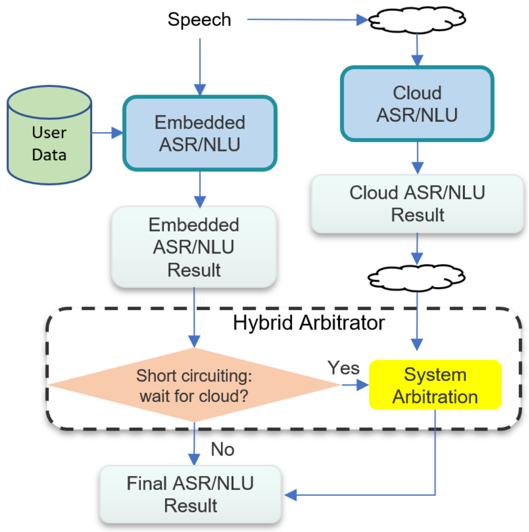

Figure 1: *Hybrid ASR Architecture*

The Hybrid Arbitrator module in Fig.1 is responsible for making two decisions:

1. Short-Circuiting – when embedded result is ready and cloud result is not, should we wait for cloud result?
2. System Arbitration – which one should we pick between embedded and cloud result?

These two questions are what this paper is going to answer. Traditionally, people primarily relied on confidence scores as the input to solve the above two problems. [9] demonstrated how the rich confidence related features could be used for arbitration. However, there are several limitations of confidence-related feature: 1. it cannot capture subtle domain and context specific differences, for example, given the same confidence input, we might still want to make different decisions depending on the context, 2. except the word level and sentence level confidence scores, most of the other confidence-related features are only available at the decoder layer, which make them hard to be accessed by the external hybrid arbitrator module, 3. it requires high-quality confidence measures, which is not always guaranteed.

In this paper, we propose that, in addition to confidence score features, we could also use ASR output, such as the top NBest word sequences from embedded and cloud, and NLU features, such as the dialogue context and recognized domain/context/action/criteria name, in the hybrid arbitration process. We demonstrate how to make use of those easily accessible information in a deep-neural-network classifier to achieve significant improvement over a confidence-score based system.

2. Hybrid Arbitration

2.1. Machine Learning Based System Arbitration

System arbitration could be modeled as a classification problem [10] with the goal to pick the result with minimum ASR errors.

2.1.1. Target Definition

We define the classification target for each utterance as the ASR result that has the lowest word error rate. However, the tricky part is, for some of the utterances, their word error rates for both embedded ASR and cloud ASR are the same. For example, both embedded ASR and cloud ASR would recognize "cancel" with very high accuracy. Depending on how we handle those utterances, we can have 4 different target definitions:

- Target ver. 1: pick embedded result as much as possible
- Target ver. 2: pick cloud result as much as possible
- Target ver. 3: Divide into 3 categories: cloud better, embedded better, or cloud and embedded work equally well
- Target ver. 4: exclude all the cases where cloud and embedded work equally well from training

2.1.2. Features sets

We use the following features from both embedded and cloud:

- ASR confidence scores, normalized to [0,1]
- NLU dialogue context, which defines what domains and contexts are supported in the current dialog stage. It is implemented as a stack of domains and contexts
- NLU recognized domain name
- NLU recognized context name
- NLU recognized action name
- NLU recognized entity names
- ASR top NBest word sequence

All the NLU features are nominal/categorical features, so we turn them into 1-hot encodings for all the machine learning approaches except DNN. In DNN approach, we notice that if we train an embedding (shared by embedded and cloud) for each of the NLU feature, the classification performance is slightly better, probably due to the input data being more compact, the model having less parameters to learn, thus harder to overfit.

ASR string feature is only used in DNN approach. We turn word sequences into local word embedding and server word embedding, with vocabulary consists of top frequent 1000 words on each ASR, an entry for out-of-vocabulary words and an entry for padding. The word embeddings are trained from scratch.

2.1.3. Classifier

We tried all the following classification techniques:

- Gaussian based Naïve Bayes classifier
- Logistic Regression
- Linear kernel and RBF kernel SVM classifier
- Random Forest with 500 Trees
- Feed-forward DNN classifier, with Relu activation on hidden layer, and Softmax activation on output layer. During training we use Adam optimizer and early stopping strategy. Fig.2 illustrates the DNN structure.

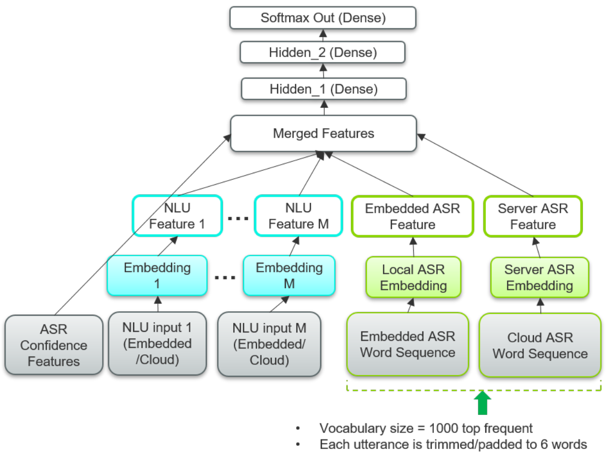

Figure 2: *DNN Architecture*

2.2. Machine Learning Based Short-Circuiting

Due to the limited computation resources, the embedded system usually have a much smaller scope and search space than its cloud side counterpart. Also, its processing is not affected by unstable internet connection. So the embedded ASR/NLU system usually returns result earlier than the cloud system. In order to generate response to end-user as soon as possible, when we are highly confident that the embedded result is accurate, we don't want to waste time waiting for cloud side result. This is the so called "short-circuiting" process – we shorten the system latency by aborting the cloud ASR/NLU process.

We model the short-circuiting process as another machine learning problem, with the optimization goal being – highest short-circuiting triggering rate while keeping the false positive rate below a threshold, such as, 0.3%. The target class is "embedded is better than or as good as cloud". Triggering rate is the percentage of the traffic that doesn't wait for cloud, and false positive means we should have waited for cloud, but we didn't.

Fig.3 illustrates how we make decision based on the confidence score of an DNN classifier, which only takes embedded features as input:

if classification confidence $\geq T$*: don't wait*
else: wait for cloud result
The threshold T is tuned based on development set.

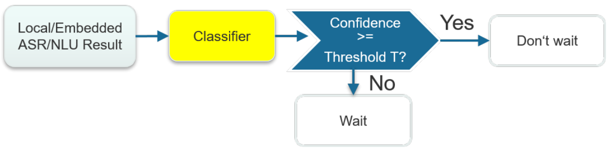

Figure 3: *Short-Circuiting Diagram*

3. Experiments

We conducted experiments on two different production system setups – system T1 and system T2. Both systems are automotive voice assistant applications. They share the same embedded ASR setup, which supports basic command and control functions. Their cloud ASR are totally different. The training data and test data are all coming from real usage logs that we randomly sampled from the production system. During the train/development/test data partitioning, we made sure all speech from the same speaker will be classified to only one set.

3.1. Production System T1

3.1.1. Experiment setup

We transcribed 0.95 million American English live speech logs, which corresponds to about 826 hours of audio from 35000 unique speakers. They were randomly sampled during Jan 2018 to Sept 2018. It is a mixture of speech from native speakers and foreign-accent speakers, and a mixture of highway condition noisy speech, local road condition noisy speech and parking lot condition speech. Out of those, we use 80% as training set, 10% as development set, and remaining 10% as test set.

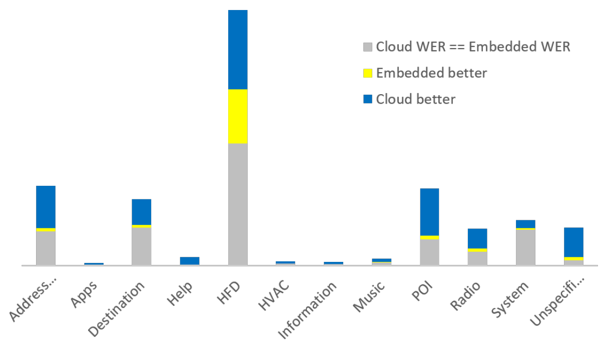

Figure 4: *Domain distribution of the T1 live log data*

Fig.4 shows the domain and arbitration target distribution of the data. Even though the cloud ASR has much bigger acoustic model and language model, it is not always the winner against embedded ASR. This is obvious in the Hands-Free-Dialing (HFD) domain, because the embedded ASR has access to end-user's personal contact book, while the cloud ASR doesn't for privacy concerns.

3.1.2. Which Target definition should we use?

For system arbitration, based on Fig.5 and Table 1, target ver. 4 works the best. However, for short-circuiting task, target ver. 1 or target ver. 3 work equally well, as seen from Fig.6.

Target Definition	Area-under-the -curve (AUC)	WER	SER
Target ver. 1	0.944	8.54	14.03
Target ver. 2	0.938	8.66	14.44
Target ver. 3	0.944	8.51	14.09
Target ver. 4	**0.953**	**8.14**	**13.11**

Table 1: *System Arbitration Performances of different Target definition, T1 system, with full features*

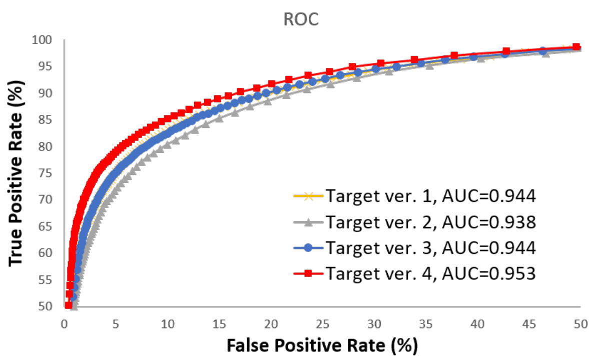

Figure 5: *Receiver operating curve (ROC) vs Target definition for System Arbitration*

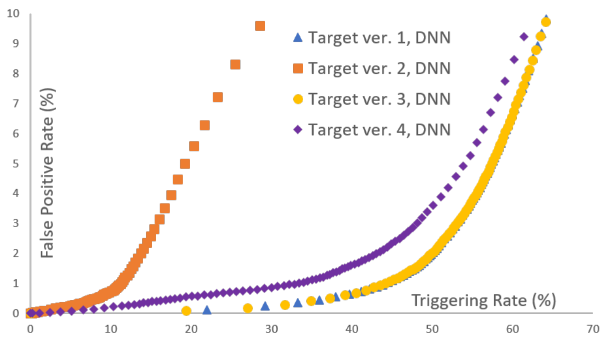

Figure 6: *Target definition vs Short-circuiting Performance*

3.1.3. System Arbitration Experiments

Table 2 compares the system arbitration performances with regards to different machine learning approaches and feature sets. DNN approach outperforms all other machine learning approaches, even with the same feature set. When we add NLU features and ASR string to the feature set, we are able to reduce final word error rate from 11.26% to 8.14%, which is quite a significant improvement. Also we can see the improvement from ”Cloud ASR only” to the best arbitration approach is very big – from a 11.73% WER to 8.14% WER, we got more than 30% relative error reduction. This justifies the importance of a high-quality hybrid arbitrator.

3.1.4. Short-Circuiting Performance

Based on Fig.7, we set the threshold T to be 0.98 for the DNN classifier with full embedded features, so that we could short-circuit/bypass about 30% of the traffic that needs to wait for cloud ASR response, with less than 0.3% mistakes. This would significantly shorten the wait time for end-users. As the Table 3 shows, the impact of 0.3% false positive rate on ASR accuracy is very small, while the 30% bypass rate led to 0.56 second latency reduction.

3.2. Production System T2

The cloud ASR in System T2 uses a much better acoustic model and language model than system T1. We collected 0.2 million transcribed American English live utterances from end-users, which corresponds to about 180 hours of audio, randomly sam-

Table 2: *System Arbitration Performances of different approaches using target ver. 4 definition, T1 system. CER stands for classification error rate, WER stands for word error rate, and SER stands for sentence error rate.*

Arbitration Approach	Features	CER	WER	SER
Embedded ASR only	N/A	78.25	49.64	46.63
Cloud ASR only	N/A	21.75	11.73	20.51
Naïve Bayes	ASR confidence feature + NLU features	19.80	11.88	18.53
SVM (Linear kernel)	ASR confidence feature + NLU features	13.82	9.57	15.93
SVM (RBF kernel, C=1)	ASR confidence feature + NLU features	13.04	9.49	15.53
Logistic Regression	ASR confidence feature + NLU features	13.40	9.46	15.59
RandomForest (#trees=500)	ASR confidence feature + NLU features	11.29	8.99	14.65
DNN	ASR confidence feature only	18.39	11.26	18.47
DNN	ASR confidence feature + NLU features	10.34	8.64	14.17
DNN	**ASR confidence feature + NLU features + ASR string**	**7.94**	**8.14**	**13.11**
Oracle (Upper bound)	N/A	0.00	5.36	9.82

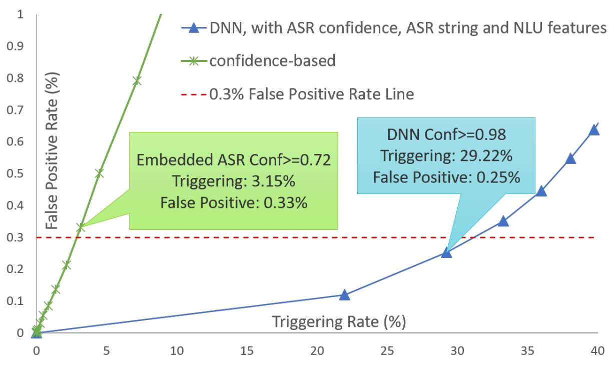

Figure 7: *Short-circuiting performance*

Table 3: *Measuring the Impact of Short-Circuiting on Hybrid Arbitration Performance*

Short-Circuiting Approach	WER	SER	Latency Reduction (s)
Confidence-Based	5.63	10.10	0.054
DNN, with all features	**5.43**	**10.00**	**0.560**
No short-circuiting	5.36	9.82	0.000

pled during Dec 2018 to Jan 2019. Out of those, we use 75% as training set, 10% as development set for hyper-parameter tuning, and remaining 15% as test set.

3.2.1. *System Arbitration Experiments*

Table 4 compares the system arbitration performances with regards to different feature sets. When we add NLU features and ASR string to the feature set, we are able to reduce final word error rate from 5.64% to 3.61%, which is significant. Again, the improvement from ”Cloud ASR only” to the best arbitration approach is very big, from a 6.68% WER to 3.61% WER, which is 46% relative error reduction.

4. Discussion

In this paper, we presented how we could use ASR word sequence and NLU information to improve hybrid arbitration performance, with the target being minimum ASR word errors.

Table 4: *System Arbitration Performances using Target ver. 4 definition, T2*

Arbitration Approach	CER	WER	SER
Embedded ASR only	82.03	48.74	45.94
Cloud ASR only	17.97	6.68	14.46
DNN with ASR confidence feature	12.32	5.64	11.22
+ NLU features	6.55	4.15	8.12
+ ASR string	**3.99**	**3.61**	**6.85**
Oracle (Upper bound)	0.00	2.57	5.12

However, what matters to the end-users ultimately is the final NLU result, and minimizing ASR word errors doesn't always lead to minimum NLU errors. For example, when cloud NLU doesn't have the access to personal data, even if the cloud ASR recognizes the speech perfectly, the cloud NLU might still fail to recognize entities, such as person names or song titles, which are coming from personal contact book and personal music library. In the future, we plan to work on hybrid arbitration targeting at minimum NLU errors.

5. Conclusions

Hybrid Arbitration is a critical problem for a lot of real-world applications. In this paper, we demonstrated how we tackle this problem using a deep-neural-network which takes ASR confidence score, ASR word sequences and NLU information as input. Both of our experiments with two production automotive voice systems show that our approach could lead to significant improvement in ASR recognition rates over cloud-only system and confidence-score based arbitration approach.

6. Acknowledgements

We would like to thank Raymond Brueckner and Daniel Willett for sharing their advices during the work.

7. References

[1] G. E. Dahl, D. Yu, L. Deng, and A. Acero, “Context-dependent pre-trained deep neural networks for large vocabulary speech recognition,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, pp. 30–42, January 2012.

[2] D. Yu and L. Deng, *Automatic Speech Recognition: A Deep Learning Approach*. Springer, 2014.

[3] D. Yu and J. Li, "Recent progresses in deep learning based acoustic models," *IEEE/CAA Journal of Automatica Sinica*, July 2017.

[4] P. Smaragdis and M. Shashanka, "A framework for secure speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1404–1413, May 2007.

[5] M. Georges, S. Kanthak, and D. Klakow, "Accurate client-server based speech recognition keeping personal data on the client," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*, 2014, pp. 3271–3275.

[6] J. G. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover)," in *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, 1997, pp. 347–354.

[7] D. Hillard, B. Hoffmeister, M. Ostendorf, R. Schlüter, and H. Ney, "irover: Improving system combination with classification," in *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, 2007, pp. 65–68.

[8] B. Hoffmeister, R. Schlüter, and H. Ney, "icnc and irover: the limits of improving system combination with classification?" in *INTERSPEECH*, 2008, pp. 232–235.

[9] K. Kumar, Z. A. Bawab, Y. Zhao, C. Liu, B. Dumoulin, and Y. Gong, "Confidence-features and confidence-scores for ASR applications in arbitration and DNN speaker adaptation," in *INTERSPEECH 2015, 16th Annual Conference of the International Speech Communication Association, Dresden, Germany, September 6-10, 2015*, 2015, pp. 702–706.

[10] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.