A NEURAL NETWORK BASED RANKING FRAMEWORK TO IMPROVE ASR WITH NLU RELATED KNOWLEDGE DEPLOYED

*Zhengyu Zhou[1], Xuchen Song[*2], Rami Botros[*3], Lin Zhao[1]*

[1] Bosch Research and Technology Center North America, Sunnyvale, CA, USA
[2] Carnegie Mellon University, Pittsburgh, PA, USA
[3] Google Inc., Mountain View, CA, USA
{Zhengyu.Zhou2, Lin.Zhao}@us.bosch.com, xuchens@andrew.cmu.edu, ramibotros@google.com

ABSTRACT

This work proposes a new neural network framework to simultaneously rank multiple hypotheses generated by one or more automatic speech recognition (ASR) engines for a speech utterance. Features fed in the framework not only include those calculated from the ASR information, but also involve natural language understanding (NLU) related features, such as trigger features capturing long-distance constraints between word/slot pairs and BLSTM features representing intent-sensitive sentence embedding. The framework predicts the ranking result of the input hypotheses, outputting the top-ranked hypothesis as the new ASR result together with its slot filling and intention detection results. We conduct the experiments on an in-car infotainment corpus and the ATIS (Airline Travel Information Systems) corpus, for which hypotheses are generated by different types of engines and a single engine, respectively. The experimental results achieved are encouraging on both data corpora (e.g., 21.9% relative reduction in word error rate over state-of-the-art Google cloud ASR performance on the ATIS testing data), proving the effectiveness of the proposed ranking framework.

Index Terms— Speech recognition, hypothesis re-ranking, joint training, slot filling, intent detection

1. INTRODUCTION

Improvements to ASR can be made in various directions, such as refining acoustic/language model [1-4] and adopting end-to-end schema [5, 6]. Among these directions, post-processing the hypotheses generated by ASR engine(s) has been a popular choice, mainly because it is much more convenient to apply linguistic knowledge to ASR hypotheses than to the decoding search space. Some post-processing methods construct certain confusion networks from the ASR hypotheses and then distinguish among competing words with the aid of acoustic/linguistic knowledge [7, 8, 9]. Many previous works rescore and rank ASR hypotheses using various advanced language models or discriminative models [10-14]. Pairwise classification based ranking approaches have also been proposed using support vector machine or neural network encoder based classifier [15, 16]. From the aspect of knowledge usage, previous ASR approaches utilize only limited linguistic knowledge, mainly modeling word sequence or extracting features directly from word sequences [11, 17]. NLU information [18-20], such as slots and intents, are typically not used in efforts to improve ASR.

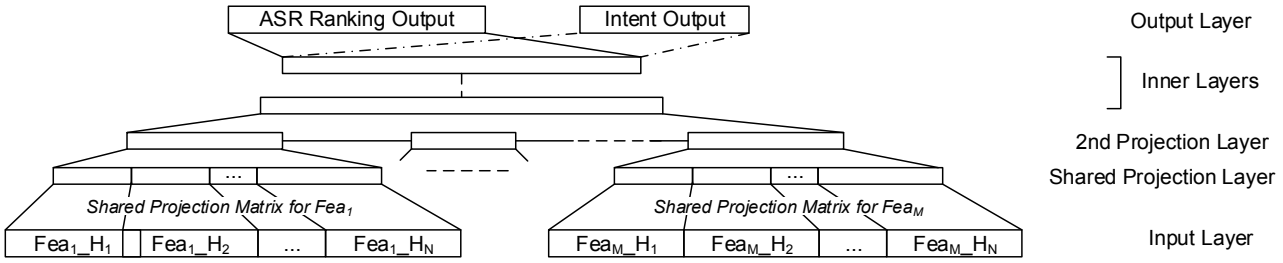
In this paper, we propose a new neural network framework to rank multiple hypotheses for one utterance. Instead of scoring each hypothesis one by one or comparing two hypotheses at a time before ranking, the framework uses all competing hypotheses as input and predicts the ranking of them simultaneously. The framework makes use of NLU knowledge to facilitate the ranking by modeling with slot/intent relevant features, and optionally joint training with intent detection. Novel soft target values that capture ranking distribution are also proposed for the training of the framework. We evaluate the framework on two data corpora in different domains, and the encouraging results obtained verify that the framework can effectively rank hypotheses generated by either a single ASR engine or multiple engines. Experimental results also show adopting NLU knowledge is beneficial, and using the soft target values instead of one-hot target values in training is important for the framework.

2. PROPOSED RANKING FRAMEWROK

2.1. Overall Framework Structure

We propose a new framework to rank ASR hypotheses. The framework is a deep feedforward neural network, which receives inputs from N (N=10 in this study) competing hypotheses generated by one or more ASR engines for a speech utterance, and predicts the ranking result for those hypotheses, optionally together with the intent detection result. The overall structure is illustrated in Figure 1.

* Xuchen Song and Rami Botros worked on this project as interns at Bosch Research and Technology Center North America, USA



Input: M different types of features (Fea_1, Fea_2,…, Fea_M) extracted from each of *N* sentence hypotheses (H_1, H_2,…, H_N)

Fig. 1. The proposed framework to rank hypotheses generated by one or more ASR engines for one speech utterance.

Features extracted from the hypotheses are fed into the input layer with the same type of features from different hypotheses concatenated together to facilitate the learning. For one feature type, hundreds or more features may be extracted from each hypothesis. We use two projection layers to handle such features. Per feature type, a shared projection matrix is first used to project the features from each hypothesis into a smaller space, and then a 2nd regular project layer is used to project those spaces from all hypotheses into an even more condense representation. The achieved representation for each type of features are then concatenated and fed into the inner layers, which are fully-connected feedforward layers. In case that a feature type may only generate one or a few features per hypothesis, such as the confidence score feature, we simply omit the projection layers for that feature type and directly feed the corresponding features extracted from all hypotheses into the inner layers, by concatenating these features with the 2nd project layer for other feature types.

The output layer contains two parts, one major part predicting the ranking results for the input hypotheses and another optional part predicting intent detection result. The major part contains *N* output nodes, which are corresponding to the *N* input hypotheses in the same order. Softmax activation [21] is used to generate the output values, and the hypotheses are then ranked based on the values accordingly. To effectively rank the hypotheses, we propose soft target values [22] (instead of one-hot values) for training as,

$$\text{target}_i = \frac{e^{-d_i}}{\sum_{i=0}^{n} e^{-d_i}} \quad (1)$$

where d_i is the Levenshtein distance of the i^{th} hypothesis from the reference sentence. With this definition, the target distribution reserves the ranking information of the input hypotheses, generating a higher score for an output node if the corresponding input hypothesis contains less ASR errors. By minimizing the Kullback-Leibler Divergence loss, the output distribution approximates the target distribution.

The "intent output" part in the output layer is optional. When intents are available, it could be beneficial to jointly train the ranking task and intent detection, since the intent information may help distinguish among the hypotheses. For the intent related output, the nodes are corresponding to possible intents, assigned with one-hot target values (1 for the reference intent and 0 for others) and trained with cross-entropy loss. When intent output is utilized, we jointly train the network, back-propagating the costs from both the ASR ranking part and the intent related part to the lower layers.

2.2. Features

In this study, four types of features (listed below) are extracted from each input hypothesis.

(1) Trigger Feature

Trigger features are used to model long/flexible-distance constraints [23]. In this work, we define triggers as a pair of linguistic units that are significantly correlated in a same sentence, where a linguistic unit could be a word or a slot (i.e., <song name>). A trigger pair (e.g., "play" → <song name>) captures the dependencies between the two units no matter how far they may be apart in a sentence. Given a collected text corpus in the domain of interest, we first process it by using the slots to replace corresponding text (e.g., using <song name> to replace "Poker Face"). We then calculate the mutual information (MI) scores of all possible trigger pairs $A \rightarrow B$ based on the corpus as follows [23],

$$MI(A:B) = P(A,B)\log\frac{P(B\mid A)}{P(B)} + P(A,\overline{B})\log\frac{P(\overline{B}\mid A)}{P(\overline{B})} + P(\overline{A},B)\log\frac{P(B\mid \overline{A})}{P(B)} + P(\overline{A},\overline{B})\log\frac{P(\overline{B}\mid \overline{A})}{P(\overline{B})} \quad (2)$$

where $\overline{A}$ / $\overline{B}$ refers to the event that *A/B* does not appear in a sentence. The top *n* trigger pairs with highest MI scores are then selected as trigger features.

When extracting trigger features from a hypothesis, a standalone NLU module is used to detect the slots in that hypothesis. The value of a trigger feature is 1 if the trigger pair appears in the hypothesis, and 0 otherwise.

(2) BOW Feature

We also adopt the bag-of-words (BOW) features, which were previously used in feedforward neural network language modeling [24]. Given a dictionary, a vector of BOW features is calculated for each hypothesis as follows,

$$bow_{decay} = \sum_{i=0}^{K} \gamma^i \vec{w}_i \quad (3)$$

where *K* is the number of words in the hypothesis and $\vec{w}_i$ is

the one-hot representation of the i-th word in the hypothesis. $\gamma \in [0,1]$ is a decaying factor, set as 0.9 as in [24].

(3) BLSTM Feature

The NLU module used in the extraction of trigger features utilizes bidirectional LSTM RNN to encode each hypothesis (see Section 2.3), where the last states of both the forward and backward RNN cover information of entire hypothesis [19]. We concatenate the two last states into a sentence embedding vector, referred to as the BLSTM features. Since the NLU module is a joint model of intent detection and slot filling, the BLSTM features are intent-sensitive.

(4) Confidence Feature

Sentence-level confidence score assigned by ASR engine to each hypothesis is also used as feature, which is directly fed into the inner layers. Note that various ASR engines may produces confidence scores using different distributions. When the input hypotheses are generated by different ASR engines, we apply a linear regression method [15] proposed previously to align the confidence scores into a same space, and then use the aligned score as the confidence feature.

2.3. A Standalone NLU Module

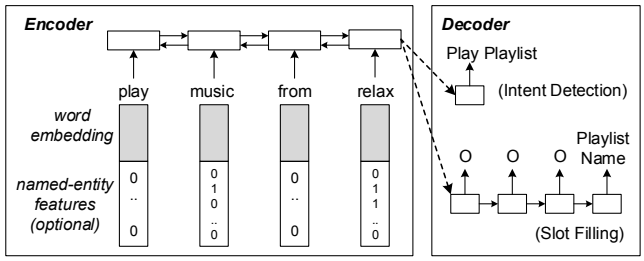


Fig. 2. Standalone encoder-decoder NLU module.

The standalone NLU module used in feature extraction as well as in the evaluation later is implemented using a state-of-the-art approach [19] that jointly models slot filling and intent detection. The module adopts a RNN based encoder-decoder structure (see Figure 2), using LSTM as the RNN unit. Pre-trained word embedding vector [25, 26] for each input word can be fed into the encoder. We further propose a method to enhance the input vector by appending named-entity features to it, when predefined name lists available. The aim is to use the added name information to facilitate learning, especially for the case when the training data is of limited size and many names occur only a few times or are unseen in it. For the named-entity features, each of them is corresponding to one name list, set as 1 if the input word is part of a name in that list and 0 otherwise. For the example shown in Figure 2, the word "relax" is both a song name and playlist name, so that the two corresponding features are set as 1 in the named-entity vector. Using this information together with the context knowledge captured by the RNN, the NLU module may identify "relax" as a playlist name even if the name "relax" is unseen in the training data.

3. EXPERIMENTS

3.1. Datasets and Experimental Settings

We use an internal in-car infotainment corpus, for which different types of ASR engines are available, to evaluate the framework for ranking hypotheses generated by multiple engines. A popular public dataset, ATIS corpus, is adopted for the task of ranking hypotheses generated by one engine.

The in-car infotainment corpus covers driver assistant related topics. The speech training/tuning/testing sets were recorded in car with relatively low noise conditions from multiple speakers with balanced gender, containing 9166, 1975, and 2080 utterances respectively. Each utterance is decoded by two domain-specific ASR engines (using grammar and statistical language model respectively, trained on separate in-domain datasets) and a general cloud engine. The three engines have complementary strengths for decoding. We feed the top-best hypothesis from each engine into the proposed framework to rank. For this corpus, most names involved are from 16 name lists, some of which are large (e.g., the song list has 5232 entries). 40 slot labels (including phone number, frequency, etc. which have no predefined list) were created following IOB schema [27] and 89 distinct intents (e.g., "tune radio frequency") are used.

The ATIS corpus [28] contains 4978 original training sentences, from which we randomly select 491 sentences as the tuning set and use the remaining as the training set. The testing set contains 893 sentences. For ATIS corpus, to simulate the condition in real-life applications, we synthesize speech utterances with noises and reverberation added, using the approach described in [29]. The state-of-the-art Google cloud ASR engine is then applied to generate 10 hypotheses at maximum with confidence scores per utterance. For ATIS data, there are 127 slot labels and 18 distinct intents.

For both in-car infotainment and ATIS data, we develop and evaluate the systems in similar ways. The standalone NLU module is trained with the reference sentences. 100d GloVe vector [25] is used as word embedding for each input word. Since ATIS data provides no name list, the named-entity vector is applied to infotainment data only, constructed based on the given name lists. The NLU module is trained in the same way as in [19], but the attention mechanism is not used due to its limited benefit observed on ASR hypotheses and efficiency consideration.

Trigger features are selected based on the training references for ATIS data, and on an additional text set of 24836 in-domain sentences for infotainment data. In this study, 850 trigger features are utilized for both corpora. For the BOW feature, the dictionary is defined as the 90% most frequent words in training references, along with an entry for out-of-vocabulary words. The alignment of confidence scores is only needed and applied for the infotainment data.

The ranking framework accepts 10 sentence hypotheses to rank. When there are less than 10 input hypotheses, we

void the space of extra hypotheses in the input layer by setting the related features as 0. In the framework, each shared projection matrix projects the corresponding features into a space of 50 nodes. This leads to a layer of size 50*10*3 (500 nodes for trigger, BOW, and BLSTM features respectively), which is further projected to a smaller 2nd projection layer (200*3 for infotainment data, 100*3 for ATIS). This layer is then concatenated with 10 confidence features to feed into input layers. For inner layers, 4 layers (500, 200, 100 and 100 nodes, respectively) are used for infotainment data, and 3 layers (200, 100 and 50 nodes, respectively) are used for ATIS, with ReLU activation [30] and batch normalization [31] applied to each layer. We adopt Adam optimization [32] to train the model in batches. Early stopping is conducted when the loss on the tuning set fails to improve for the last 30 iterations. The model having the best performance on tuning data is used in evaluation.

3.2. Experimental Results on In-Car Infotainment Data

For the standalone NLU module, we found that feeding the named-entity features into the encoder is beneficial, e.g., reducing the intent detection error rate from 9.12% to 5.17% and raising the slot filling F1 score from 64.55 to 90.68 on testing references. This indicates that the name information introduced effectively relieves the difficulty in learning when using large name lists and limited training data.

For the ranking framework, we first train a framework using only ASR ranking output, referred to as the ASR-alone framework, and then train a joint framework using both ASR ranking and intent outputs. The evaluation results of the two frameworks are included in Table 1.

Table 1. Results on in-car infotainment testing data.

	WER%	Intent Error%	Slot F1
Oracle hypo. +NLU	3.87	7.84	84.51
Top-scored hypo. +NLU	7.56	11.37	79.73
ASR-alone Framework	7.05	10.49	79.82
Joint Framework	**6.69**	**10.00**	**80.50**

In Table 1, "Oracle hypo." and "Top-scored hypo." refer to the hypothesis with the lower word error rate (WER) and highest (aligned) confidence score, respectively, among competing hypotheses. "+NLU" denotes the procedure that applies the standalone NLU module to a hypothesis to get NLU results (i.e., slot filling F1 score and intent detection error rate). We evaluate both WER and NLU results because for the ranking framework, each input hypothesis is processed by the NLU module during feature extraction, obtaining its NLU results. When the framework predicts the top-ranked hypothesis, it also retrieves the NLU results associated with that hypothesis. Note that for the joint framework, the intent related output also predicts an intent. However we notice that this predicted intent performs worse than the intent assigned to the top-ranked hypothesis, possibly due to the confusion introduced by competing hypotheses. We thus choose the later as the intent result.

Table 1 shows that the ASR-alone framework brings a 6.75% relative reduction in WER over the "top-scored hypo." baseline (i.e., the performance of ranking hypotheses based on aligned confidence score only), which outperforms each individual engine's performance. The joint framework enlarges the benefit to 11.51% relative WER reduction. For NLU results, similar improvements are achieved.

Experiments also show that for the framework, adopting the proposed soft target values for ranking is important. For example, when replacing the soft target values with one-hot values, the WER obtained by the joint model rises to 7.21%.

We also observe that all the four types of features are beneficial for the in-car infotainment data, deleting each one will lead to worse performances. For example, removing the slot-based trigger features from the joint framework increases the WER of the resulting model to 7.28%.

3.3. Experimental Results on ATIS Data

On ATIS data, all types of features are observed beneficial except BLSTM feature, possibly because the hypotheses are generated by one engine and are highly similar in sentence embedding. We thus remove the BLSTM features from the ranking frameworks and the results are shown in Table 2. We can see that the ASR-alone framework achieves the best performance, obtaining 21.9% relative reduction in WER over state-of-the-art Google ASR performance (i.e., "top-scored hypo."). Joint training with intent detection brings no improvement in this case, also because of the high similarity among the hypotheses, which typically bear the same intent. The improvements achieved on ATIS are bigger than those on the infotainment data, possibly because more hypotheses with lower oracle performance are used in ranking for ATIS. For ATIS data, adopting the soft target values is critical. Using one-hot target values instead in the training of ASR-alone framework increases the resulting WER to 6.68%.

Table 2. Results on ATIS testing data.

	WER%	Intent Error%	Slot F1
Oracle hypo. +NLU	1.24	2.58	94.05
Top-scored hypo. +NLU	6.53	3.92	90.89
ASR-alone Framework	**5.10**	**3.25**	**92.50**
Joint Framework	5.51	3.36	91.93

4. CONCLUSION

In this paper, we proposed a novel network framework to rank competing hypotheses for a speech utterance, which is shown effective no matter the hypotheses are generated by multiple ASR engines or one engine. The framework is trained with soft target values, using not only ASR related but also NLU related features. Joint training the framework with intent detection is found beneficial when the hypotheses come from different types of engines. The framework can be further refined/improved in various directions, such as introducing new features and improving the network design.

5. REFERENCES

[1] G. Hinton, et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, 29.6: 82-97, 2012.

[2] A. Zeyer, et al., "A comprehensive study of deep bidirectional LSTM RNNs for acoustic modeling in speech recognition," *Proc. ICASSP*, 2017.

[3] T. Mikolov, et al., "Recurrent neural network based language model," *Proc. INTERSPEECH*, 2010.

[4] R. Jozefowicz, et al., "Exploring the limits of language modeling," arXiv preprint arXiv:1602.02410, 2016.

[5] Y. Zhang, et al., "Very deep convolutional networks for end-to-end speech recognition," *Proc. ICASSP*, 2017.

[6] C.C. Chiu, et al., "State-of-the-art speech recognition with sequence-to-sequence models," *Proc. ICASSP*, 2018.

[7] J. Fiscus, "A post-processing system to yield reduced error word rates: Recognizer output voting error reduction (ROVER)," *Proc. ASRU*, pp. 347–354, 1997.

[8] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: Word error minimization and other applications of confusion networks," C*omputer Speech and Language*, vol. 14, no. 4, pp. 373–400, Oct. 2000.

[9] Z. Zhou, *An Error Detection and Correction Framework to Improve Large Vocabulary Continuous Speech Recognition*, PhD Thesis, 2009.

[10] Y. Huang, et al., "Whole Sentence Neural Language Models," *Proc. ICASSP*, 2018.

[11] T. Mikolov, et al., "Empirical evaluation and combination of advanced language modeling techniques," *Proc. INTERSPEECH,* 2011.

[12] Z. Zhou, et al., "A comparative study of discriminative methods for reranking LVCSR n-best hypotheses in domain adaptation and generalization," *Proc. ICASSP*, 2006.

[13] E. Arisoy, et al., "Discriminative language modeling with linguistic and statistically derived features," *IEEE Tran. on Audio, Speech, and Language Processing,* 20(2), pp.540-550, 2012.

[14] Z. Zhou and H. Meng, "Pseudo-conventional n-gram representation of the discriminative n-gram model for LVCSR," *IEEE Journal of Selected Topics in Signal Processing*, 4(6), pp.943-952., 2010.

[15] Z. Zhou and Z. Feng, "System and method for speech recognition," U.S. Patent 9,959,861, 2018.

[16] A. Ogawa, et al., "Rescoring N-best speech recognition list based on one-on-one hypothesis comparison using encoder-classifier model," *Proc. ICASSP*, 2018.

[17] J.T. Goodman, "A bit of progress in language modeling," *Computer Speech & Language*, 15(4), pp.403-434, 2001.

[18] D. Jurafsky and J.H. Martin, "Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition," P*rentice Hall series in artificial intelligence*, pp.1-1024, 2009.

[19] B. Liu and I. Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling," arXiv preprint arXiv:1609.01454, 2016.

[20] K. Hashimoto, et al., "A joint many-task model: Growing a neural network for multiple NLP tasks," arXiv preprint arXiv:1611.01587, 2016.

[21] Y. LeCun, L. Bottou, and Y. Bengio, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[22] Z. Zhou and R. Botros, "System and method for ranking of hybrid speech recognition results with neural networks," U.S. Patent Application 15/353,767, 2018.

[23] R. Rosenfeld, *Adaptive Statistical Language Modeling: A Maximum Entropy Approach*, PhD Thesis, 1994.

[24] I. Kazuki, R. Schlüter and H. Ney, "Bag-of-words input for long history representation in neural network-based language Models for speech recognition," *Proc. INTERSPEECH*, 2015.

[25] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," *Proc. EMNLP,* 2014. https://nlp.stanford.edu/projects/glove/

[26] A. Joulin, et al., "Fasttext. zip: Compressing text classification models," arXiv preprint arXiv:1612.03651, 2016.

[27] L.A. Ramshaw, L.A. and M.P. Marcus, "Text chunking using transformation-based learning," *Natural language processing using very large corpora*, pp. 157-176, 1999.

[28] T. Charles, et al., "The ATIS spoken language systems pilot corpus," *Proc. HLT*, 1990.

[29] R. Schumann and P. Angkititrakul, "Incorporating ASR Errors with Attention-Based, Jointly Trained RNN for Intent Detection and Slot Filling," *Proc. ICASSP*, 2018.

[30] A. Krizhevsky, I. Sutskever, and G.E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, pp. 1097-1105, 2012.

[31] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," arXiv preprint arXiv:1502.0316, 2015.

[32] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.