THE NAIST ASR SYSTEM FOR THE 2015 MULTI-GENRE BROADCAST CHALLENGE: ON COMBINATION OF DEEP LEARNING SYSTEMS USING A RANK-SCORE FUNCTION

Quoc Truong Do, Michael Heck, Sakriani Sakti, Graham Neubig, Tomoki Toda, Satoshi Nakamura

Augmented Human Communication Laboratory,
Graduate School of Information Science,
Nara Institute of Science and Technology,
Nara, Japan

{do.truong.dj3|michael-h|ssakti|neubig|tomoki|s-nakamura}@is.naist.jp

ABSTRACT

The Multi-Genre Broadcast challenge is an official challenge of the IEEE Automatic Speech Recognition and Understanding Workshop. This paper presents NAISTs contribution to the premiere of this challenge. The presented speech-to-text system for English makes use of various front-ends (e.g., MFCC, i-vector and FBANK), DNN acoustic models and several language models for decoding and rescoring (N-gram, RNNLM). Subsets of the training data with varying sizes were evaluated with respect to the overall training quality. Two speech segmentation systems were developed for the challenge, based on DNNs and GMM-HMMs. Recognition was performed in three stages: Decoding, lattice rescoring and system combination. This paper focuses on the system combination experiments and presents a rank-score based system weighting approach, which gave better performance compared to a normal system combination strategy. The DNN based ASR system trained on MFCC + i-vector features with the sMBR training criterion gives the best performance of 27.8% WER, and thus significantly outperforms the baseline DNN-HMM sMBR yielding 33.7% WER.

Index Terms— speech recognition, ASRU MGB, broadcast, evaluation system, system development

1. INTRODUCTION

This paper describes NAISTs contribution to the Multi-Genre Broadcast (MGB) challenge[1], an evaluation campaign for state-of-the-art automatic speech recognition systems that need to be capable of reliably transcribing audio from a multi-genre, multi-topic and multi-channel domain. Fixed sets of audio and text data for training were provided, as well as a pronunciation lexicon, which allows for a comparison of multiple systems on matching data and conditions. Participants were free to join any of the four provided tasks. NAIST entered the challenge by participating in the *speech-to-text transcription of broadcast television* task, which is a standard speech transcription task, where systems have to operate on a diverse collection of BBC television shows. The evaluation and the individual tracks are described in detail in [1]. Our contribution to the challenge comes in the form of an English speech recognition system that has been built from scratch, utilizing the Kaldi speech recognition toolkit [2].

The developed speech-to-text system makes use of various front-ends, deep neural net (DNN) acoustic models and several language models for decoding and rescoring (see Subsections 2.1 and 2.2, respectively). This paper also discusses several approaches to automatic segmentation (see Section 4), as one non-mandatory sub-task of the challenge was to automatically obtain utterance time stamps prior to decoding.

System combination is a common approach for high performance speech recognition, especially if recognition in real-time is not the major concern. Recognizer output voting error reduction (ROVER) [3] and confusion network combination (CNC) are among the most popular methods. ROVER generally allows for some form of weighting with the help of confidence scores, which is inherent to CNC. Studies like [4] and [5] affirm the advantages of confidence based weighting strategies. However, it is common practice that systems which contribute to a combination do so with equal shares: Besides word or segment based weighting, systems contribute equally to the final output. This strategy however might fail in cases where system performances are unbalanced. The better hypotheses might simply be overpowered by suboptimal alternatives.

We investigated the potentially positive effects of weighted system combination that makes use of weights on system level. Our weighting scheme is data driven and utilizes a rank-score: Each system, according to its rank when judged by its recognition accuracy on a development set, contributes to the combination with an individual weight. System weights are correlated to the rank-scores of the systems. The experimental results show a consistently better performance compared to normal system combination strategies. We were able to reproduce the findings of [4] that standard and confidence-weighted ROVER on two systems is outperformed by lattice-based combination strategies, but prevails for combinations of more than two systems. Moreover, we were able to observe that the benefits by system-based weighting are significantly higher for ROVER than for the lattice-based combination alternatives, so that ROVER even has the potential to outperform the latter. Our best combination resulted in a WER of 27.5% on the development set. A detailed description of our approach is found in Subsection 2.3, and the results of the combination experiments are described in Subsection 5.3.

2. OVERALL SYSTEM

In this section, we describe the acoustic, language model training, and also system the system combination strategy. An overview of our system is illustrated in Figure 1.

[1] www.mgb-challenge.org

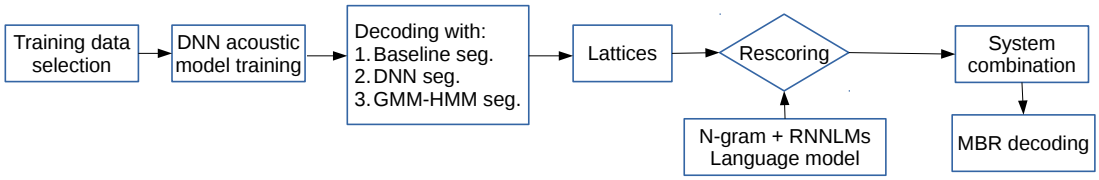

Fig. 1. System overview

2.1. Acoustic model training

We tested several acoustic model training strategies during development. Deep neural network acoustic models trained with 2 different types of input features were adopted, where one are either standard MFCC + i-vector or FBANK + i-vector features. The DNN with standard MFCC input features is trained by the cross-entropy criterion, whereas the DNN with standard feature (MFCC, FBANK) + i-vector features is a p-norm deep neural network [6], which is trained with both cross-entropy (CE) and state-level minimum Bayes risk (sMBR) training criterion [7, 8]. The full training procedure is illustrated in Figure 2.

2.1.1. I-Vector extraction

The i-vector was initially introduced for speaker recognition tasks [9], and recently has drawn researcher attention in the field of speech recognition. The i-vector $\mathbf{w}$ is defined in the context of the following term,

$$\mathbf{M} = \mathbf{m} + \mathbf{T}\mathbf{w}, \tag{1}$$

where $\mathbf{M}$ is the utterance supervector which depends on speaker and channel dependent components [10], $\mathbf{m}$ is the mean supervector of a universal background model (UBM), $\mathbf{T}$ is low rank rectangular total variability matrix, and $\mathbf{w}$ is the i-vector following the standard normal distribution $N(0, I)$.

Given input feature frames $\mathbf{Y}$, the i-vector $\mathbf{w}$ can be defined by the mean of the posterior distribution $P(\mathbf{w}|\mathbf{Y})$, where this posterior distribution is a Gaussian distribution [11].

The matrix $\mathbf{T}$, which refers to the i-vector extractor, is trained for every second feature frame to speed up the training. The i-vector extractor and the UBM models used for the experiments described in this paper were trained using 338 hours of training data including 48,587 speakers. The UBM model has 512 Gaussian components. As mentioned before, two different types of speech features (MFCC and FBANK) were utilized.

2.1.2. DNN cross-entropy

The first DNN model can be considered a standard DNN acoustic model with 6 hidden layers, where each layer consists of 2048 nodes. The non-linear sigmoid activation function is applied in each hidden layer, and the softmax function is applied in the output layer. The input features are LDA + MLLT + fMLLR on top of MFCC. The feature frames are also spliced with 5 preceding and 5 succeeding frames, resulting in the final 440 dimensional DNN input feature vector covering 11 frames of context.

First, we performed the pre-training with deep belief network (RBM) [12]. After that, the DNN was trained using the back-propagation algorithm and stochastic gradient descent with frame cross-entropy (CE) criterion as implemented by the Kaldi speech recognition toolkit [2].

2.1.3. P-norm DNN

As a second type of model, the p-norm deep neural network [6] was adopted. The p-norm is a "dimension-reducing" non-linearity that is inspired by maxout

$$y = ||\mathbf{x}||_p = \left(\sum_i |x_i|^p \right)^{1/p}, \tag{2}$$

where here the vector $\mathbf{x}$ represents a bundled set of 10 feature vectors, p is the normalized parameter and is set to 2 as it showed the best performance as described in [6]. The number of hidden layers is 6. The 40 dimensional MFCC or FBANK feature vectors and the 100 dimensional i-vectors are stacked to form a 140 dimensional DNN input feature.

The parameters are trained by using either CE or sMBR criterion as implemented in Kaldi. For each type of input features, two DNN models are trained, one by the CE criterion, the other by the sMBR criterion. After decoding, the decoding lattices of both systems are combined to produce the final decoding lattices.

2.2. Language model

2.2.1. N-gram

N-grams have long been a standard language modeling technique for ASR, where $N-1$ words are used as context to predict the next word. The larger the context, the more data is required to avoid the data sparsity problem. During the experiments described here, two N-gram language models were trained with Kneser-Ney smoothing [13] implemented in SRILM language modeling [14], a 3-gram LM pruned with probability 10^{-8} for decoding purposes, and a full 4-gram model for rescoring in a second pass. All available textual training data was used for the training of these models.

2.2.2. RNNLMs

Recurrent neural network language models have shown to have an advantage over the standard N-gram language model. There are several reasons for this, perhaps the most notable being that RNNLMs can capture the context of entire utterances, which is difficult to do in with standard N-grams. [15, 16] have also shown that RNNLMs can significantly improve the performance of speech recognition, especially when RNN models are interpolated with N-gram language models. However, the drawback of RNNLMs is the computational complexity. Therefore, this type of language model is usually used for rescoring in two-pass decoding systems.

The systems that we developed for the MGB challenge adopt a class-based RNNLM [15], whose 3-layer topology is illustrated in Figure 3. The hidden activation values of previous word tokens are concatenated with the current word vector to form a new input

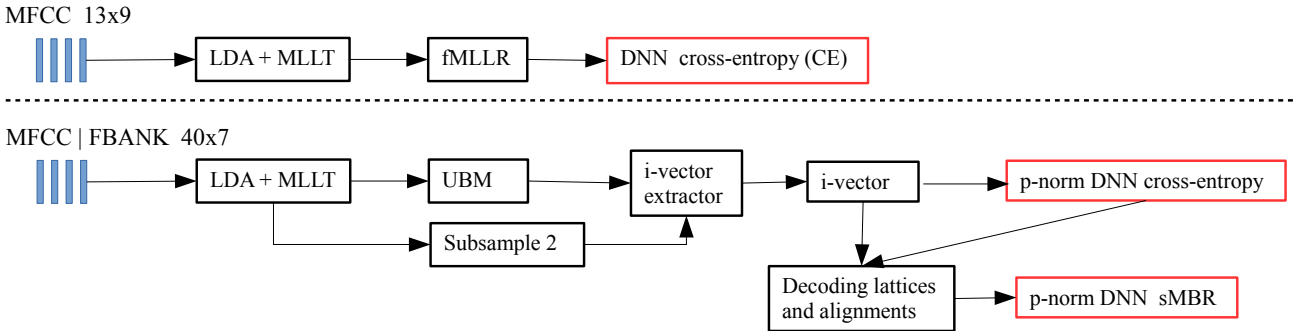

Fig. 2. Acoustic model training procedure.

vector for the network. The probability of w_i, which is the i^{th} word belonging to class c_i is calculated as

$$P(w_i|\mathbf{x}_i) = P(w_i|c_i, \mathbf{s}_i)P(c_i|\mathbf{s}_i) \quad (3)$$

The RNNLM is comprised of 150 hidden nodes and 400 classes. The model is trained using the threaded version of the RNNLM toolkit. It took about 5 days to finish the training process.

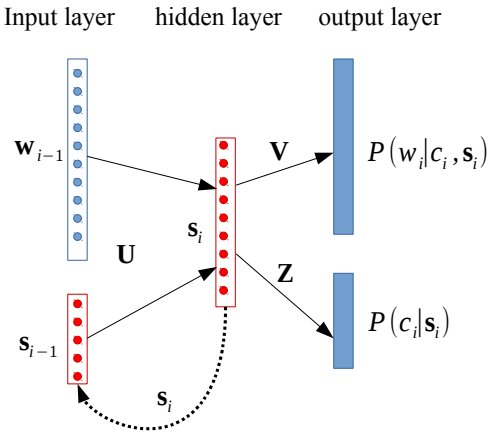

Fig. 3. RNNLMs topology

2.3. Rank-score based combination

One focus of these experiments was on the introduction of a weighting scheme which takes into consideration the potency of the contributing systems. A straightforward and widely used approach is to employ equal weights to all system outputs. However, in a scenario where various systems differ greatly in performance, this strategy might nullify the positive effects of hypothesis combination or even lead to a degradation in accuracy, as errors introduced by weaker systems might overpower the correct hypotheses of the more accurate systems. This effect was indeed observable during combination experiments with equally weighted contributions.

In order to maximize the positive effects of combination, we introduced a method for output weighting on system level. Our weighting scheme is data driven and makes use of a rank-score: Each system, according to its rank when judged by its recognition accuracy on a development set, contributes to the combination with an individual weight. System weights are conditioned to the rank-score of the systems. Let $\mathrm{rank}(s_n) \in \{1, \ldots, |\mathcal{S}|\}$ be the rank of a system $s_n \in \mathcal{S}$, where the system s^* with the highest accuracy $\mathrm{acc}(s^*)$ has rank 1. The rank-score of a system s_n is

$$\mathrm{acc}(s_n) \cdot (|\mathcal{S}| + 1 - \mathrm{rank}(s_n)) \quad (4)$$

A numerically lower rank indicates a system with higher performance. We weighted the individual systems during lattice combination according to:

$$\mathrm{weight}(s_n) = \frac{\mathrm{acc}(s_n) \cdot (|\mathcal{S}| + 1 - \mathrm{rank}(s_n))}{\sum_{s_n \in \mathcal{S}} \mathrm{acc}(s_n) \cdot (|\mathcal{S}| + 1 - \mathrm{rank}(s_n))} \quad (5)$$

For ROVER, an approximate method was used due to the fact that direct weighting was not possible. Instead, systems were taken into consideration multiple times for the combination, according to their respective ranks: In a combination of 4 systems, the best system enters ROVER 4 times, the second best 3 times and so on.

3. TRAINING DATA

3.1. Original data

As described in [1], the original data is created via a two-step refinement, where the original transcription from BBC is aligned to the audio and a score for data selection is computed for quality measurement. After refinement, a total of 1005 hours of training data is released along with the score for each segment. Phone matched error rate (PMER) and word matched error rate (WMER) range from 0 to 100.

3.2. Data selection

We performed data selection to maximize the use of the provided training material. The procedure shall be elaborated briefly. The purpose was to determine the most suitable data subset for acoustic model training.

Data was selected according to the WMER measure. For experiments, 4 sub data sets were selected based of WMER 10, 15 and 20. Then, a GMM-HMM speaker adapted model for each data set was trained. Finally, held-out data from the development set covering 2000 utterances was decoded to calculate the WER. The results are listed in Table 3.2.

As a result, although the difference is small one can see that the data set with a WMER of 15 gives the best results in terms of WER. The size of the data is approximately 338 hours, which is also an

appropriate amount of data to train the acoustic model. This data was used for the acoustic model training for all further experiments.

WMER	Size (hours)	WER
10	240	63.70
15	338	63.12
20	400	63.15

Table 1. The WER of GMM-HMM models trained on different data sub-sets.

4. AUTOMATIC SEGMENTATION

The evaluation set was provided with an automatically generated segmentation by the organizers. However, participants were encouraged to produce their own segmentations, i.e., to find means to automatically write utterance time stamps given the raw audio data. We evaluated the effectiveness of two different approaches to automatic segmentation of audio data, which are:

a) *NN based* segmentation on feature vectors, followed by smoothing. A deep neural network was trained to perform a 2-class classification of input feature vectors into frame based speech and non-speech tokens. Smoothing is performed by merging neighboring features across their respective classes, where tolerance parameter for noise enclosure allows for optimizing towards less fractured utterance segments. Padding is applied to cushion the sharp decisions that are inherent to the frame based classifier. *b*) *GMM based* segmentation using a speech and a non-speech model. This method uses a Viterbi based decoder and GMM-HMM models to classify consecutively observed feature vectors into the two broad sound categories. The mechanics of the general framework are comparable to the one presented in [17]. Padding is applied during a post-processing step.

4.1. NN based segmentation

The NN based segmentation framework makes use of a deep feed-forward neural net to perform a binary classification of the feature vectors into speech or non-speech. The neural net is comprised of 5 hidden layers, where the first 4 layers are composed of 1024 nodes each, and the 5th layer has 512 nodes. The non-linear tanh function is used as an activation function for each hidden layer as well as the output layer. To avoid the over-fitting problem, the L_2 regularization method has been adopted, with L_2 set to 0.0001. The neural net is trained using the standard back-propagation algorithm. The learning rate is initialized at 0.1 and scaled by 0.5 when the validation error reduction between two consecutive epochs is less than 0.05. The training process will stop when the learning rate falls below a value of 0.0002.

4.2. GMM based segmentation

The GMM segmenter is essentially a speech recognizer that is capable of discriminating two classes of sounds. This system has been built with the Janus speech recognition toolkit [18]. Segments are modeled as single HMM states, where the minimal segment lengths are directly modeled by the HMM topology. Each GMM consists of 128 Gaussians. The dimension is given by the input feature vector type. The acoustic model is trained according to the maximum likelihood criterion, where the GMMs grow incrementally in several iterations of "split-and-merge" training [19].

4.3. Segmentation Data Selection

Due to the fact that the provided acoustic training data did not come with labels for noise in the recordings, the assumption "everything in between annotated speech parts is likely to be noise" was made. Thus, speech samples were extracted from audio segments annotated as speech, and non-speech samples were extracted from the gaps in between speech segments. In contrast to the acoustic model training for the ASR systems, this extraction was done on the full set of provided training data, as the timing informations had to be complete in order to most likely only extract non-speech from un-annotated audio segments. Only a random subset of the overall data was used for training, as preliminary tests showed that an exhaustive use of the data was not resulting in better segmentations. Ultimately, each segmenter described here was trained on 38 hours of speech data, and 26 hours of non-speech data randomly selected from the training data.

4.4. Development

During system development, various feature vector types were utilized for both segmentation approaches. Besides 13 dimensional standard MFCC features, stacked MFCCs with a context of 2 (the 2 preceding and 2 succeeding vectors are stacked to the center vector, here denoted as MFCC-2) as well as 42 dimensional LDA transformed MFCCs after stacking with a context of 7 were taken into consideration for a cross-feature and cross-framework comparison on segmentation accuracy. The respective contexts proved to be good choices during preliminary experiments. Future experiments might be conducted for optimizing those parameters. For evaluating segmentation accuracy, the provided segmentation was considered the ground truth. Segment coverage was computed on frame level and evaluated in terms of accuracy and receiver operating characteristic. System parameters such as the padding factors were tuned on the development set. Table 2 lists the performance of all configurations on the official challenge development set.

Segmentation	Feat	ACC	TP	TN
NN	MFCC	70.6%	89.5%	33.1%
	MFCC-2	71.5%	89.3%	36.3%
	LDA	73.7%	86.2%	49.0%
GMM	MFCC	71.5%	85.1%	44.4%
	MFCC-2	71.6%	90.8%	33.7%
	LDA	76.6%	93.0%	44.1%

Table 2. Segmentation accuracy (ACC), true positive (TP) rate and true negative (TN) rate of the individual models and front-ends (Feat), computed on the challenge development set given the provided segmentation.

Albeit all systems showing a rather poor accuracy, the GMM based system with LDA front-end provided the most accurate segmentation results, and also gained the highest TP rate while still maintaining a competitive TN rate, given the alternative systems. The contrastive submission makes use of this segmentation approach.

5. ASR PERFORMANCE

We built several systems for the evaluation, featuring various front-ends, acoustic model types and training criteria. The training data for acoustic modeling was fixed on one subset of the provided material prior to system development. Likewise, all systems are based on the same phoneme set and language model training data.

System	Features	Criterion	Activation function	Rescoring	
				4-gram LM	RNNLM
1	MFCC	cross-entropy	sigmoid	37.1%	36.2%
2	FBANK + i-vector	cross-entropy	p-norm	30.8%	30.1%
3	MFCC + i-vector	cross-entropy	p-norm	30.1%	29.3%
4	MFCC + i-vector	sMBR	p-norm	28.5%	27.8%

Table 3. Individual system performances of all DNN based recognizers in WER after rescoring.

Systems				Rescoring	
1	2	3	4	4-gram	RNNLM
✓	✓	✓	✓	28.4%	28.0%
✓		✓	✓	28.6%	28.4%
	✓	✓	✓	28.3%	28.0%
	✓		✓	28.1%	27.9%
		✓	✓	28.6%	28.1%

Table 4. Performance of the lattice combined systems in WER, when applying equal weights for all systems.

Systems				Rescoring	
1	2	3	4	4-gram	RNNLM
✓	✓	✓	✓	28.7%	28.0%
✓		✓	✓	28.7%	28.0%
	✓	✓	✓	28.5%	27.8%
	✓		✓	28.9%	29.5%
		✓	✓	30.9%	30.5%

Table 5. Performance of the ROVER combined systems in WER, when applying equal weights for all systems.

5.1. Decoding

First, a GMM-HMM based system was trained, which, in addition to providing a baseline, was the basis for all subsequent DNN based systems. Of these, four types were trained, each utilizing one distinct input feature type: MFCC, MFCC + i-vector, and FBANK + i-vector, where the model for the second feature type was trained with two different training criteria (see section 2.1). Decoding was performed with the pruned 3-gram language model, followed by lattice rescoring with a 4-gram and RNNLM language model.

5.2. Single system performance

Table 3 lists the performances of all single systems on the development set. By comparing systems 2 and 3, it can be seen that the MFCC + i-vector feature vectors lead to a better performance than FBANK + i-vector. The results also reveal that the systems using the combination of standard features and i-vector features outperforms the one using the standard MFCC features alone. Due to the relatively poor performance of the GMM-HMM system compared to the DNN systems, these numbers were omitted from the table.

The primary submission is the decoding result of the DNN i-vector trained with the sMBR criterion after rescoring with the 4-gram LM. This is due to unfinished RNNLM rescoring and system combination at the time of the final submission date. The contrastive submission is generated by the same system, but on the custom segmentation provided by the GMM based framework presented in Section 4.

Systems				Rescoring	
1	2	3	4	4-gram	RNNLM
0.09	0.20	0.30	0.41	28.2%	27.9%
0.15		0.34	0.51	28.3%	28.0%
	0.16	0.33	0.51	28.2%	27.8%
	32.9		67.1	27.9%	27.8%
		32.9	67.1	28.4%	27.9%

Table 6. Performance of the combined systems in WER, when applying the rank-score based weighting.

Systems				Rescoring	
1	2	3	4	4-gram	RNNLM
1	2	3	4	28.2%	27.5%
1		2	3	28.1%	27.6%
	1	2	3	28.3%	27.5%
	1		2	28.4%	27.7%
		1	2	28.4%	27.7%

Table 7. Performance of the ROVER combined systems in WER, when applying the rank-score based weighting.

5.3. System combination performance

We evaluated different combination methods using ROVER and decoding lattices, as well as system combination with the rank-score function.

Tables 4 and 5 show the results of lattice combination and ROVER respectively, when using equal weights. With lattice combination, it was possible to produce slight improvements for the N-gram rescored outputs, exceeding the performance of the best single system. System combination after RNNLM rescoring however did not lead to any improvements that would outperform the best single system. This might be due to the nature of the lattice output after RNNLM rescoring. The pre-selection of n-best lists for each system output might limit the possibilities of system combination to further enhance the decoding results. The ROVER combination was not able to introduce improvements to the final hypothesis, when only two systems were contributing to the combination. When multiple systems were combined, no ROVER combination could outperform the best single system after 4-gram rescoring and RNNLM rescoring, respectively. However, our numbers affirm the observations made in [4], that ROVER has the potence to outperform lattice based combination, when more than two systems are fused together.

Table 6 lists the performance of lattice combination using the rank-score based weighting method from Subsection 2.3. The first four columns list the individual system weights. Weights were computed separately for both rescoring types, however, they happen to be entirely identical, thus the compressed way of presentation. The consistent decrease in WER for all tested combinations proof that rank-based weighting reliably improves system combination efficiency.

Taking into consideration all available systems, performance can be improved from 28.5% (highest scoring single system) to 28.2% after N-gram LM based rescoring. By limiting the combination to two specific systems, WER further drops to 27.9%. Upon closer inspection this seems reasonable, as the two systems involved utilize fairly different input features (FBANK + i-vector and MFCC + i-vector) and training criteria (cross-entropy and sMBR), but both perform comparatively well, so that combination has the potential to affect the overall outcome of the decoding.

For ROVER, implicit weighting was not possible. To simulate the effect, hypotheses were taken into consideration multiple times for the combination, according to their respective ranks, using the method described in Subsection 2.3. As can be seen in Table 7, ranked ROVER can achieve the same performance as lattice combination after N-gram rescoring, and moreover clearly outperforms lattice combination after RNNLM rescoring, pushing the optimal recognition performance after combination to 27.5% WER.

It is noteworthy that for all presented systems in this section, minimum Bayes risk (MBR) [20] decoding given the combined lattices was applied to minimize the expected WER.

6. CONCLUSION

This paper described the structure and development of NAISTs English ASR system for the ASRU MGB 2015 challenge. Different architectures of deep neural network based models as well as various types of input features such as MFCC, FBANK and i-vector have been evaluated. The results show that the p-norm DNN trained on combined MFCC + i-vector feature vectors following the sMBR training criterion gives the best performance for a single system, achieving 27.8% WER. This significantly outperforms the baseline system which yields 48.5% WER. The results allow the observation that standard system combination helps to reduce the WER on N-gram rescored lattices. The potence of system combination can be significantly enhanced by using a rank-score based weighting on system level, where systems contribute to the combination in accordance to their individual performance. In doing so, we were able to improve combination results on N-gram rescored lattices. Where lattice rescoring was not able to improve on RNNLM rescored lattices any further, ranked ROVER pushed the performance by recombination on word level, achieving 27.5% WER, which was the highest value we achieved during these experiments.

7. ACKNOWLEDGEMENTS

Part of this work was supported by JSPS KAKENHI Grant Number 24240032 and by the Commissioned Research of National Institute of Information and Communications Technology (NICT), Japan.

8. REFERENCES

[1] P Bell, MJF Gales, T Hain, J Kilgour, P Lanchantin, X Liu, A McParland, S Renals, O Saz, M Wester, and PC Woodland, "The MGB Challenge: Evaluating multi-genre broadcast media recognition," in *Procedings of ASRU*, 2015.

[2] D. Povey, A. Ghoshal, G. Boulianne, N. Goel, M. Hannemann, Y. Qian, P. Schwarz, and G. Stemmer, "The Kaldi speech recognition toolkit," in *Proceedings of IEEE workshop*, 2011.

[3] J.G Fiscus, "A post-processing system to yield reduced word error rates: Recognizer Output Voting Error Reduction (ROVER)," in *Proceedings of ASRU*, Dec. 1997, pp. 347–354.

[4] B. Hoffmeister, D. Hillard, S. Hahn, R. Schlüter, M. Ostendorf, and H. Ney, "Cross-site and intra-site ASR system combination: Comparisons on lattice and 1-best methods.," in *Proceedings of ICASSP*, 2007, pp. 1145–1148.

[5] K. Audhkhasi, A. M. Zavou, Panayiotis G. Georgiou, and S. Narayanan, "Empirical link between hypothesis diversity and fusion performance in an ensemble of automatic speech recognition systems.," in *Proceedings of INTERSPEECH*, 2013, pp. 3082–3086.

[6] Xiaohui Z., J. Trmal, D. Povey, and S. Khudanpur, "Improving deep neural network acoustic models using generalized maxout networks," in *Proceedings of ICASSP*, May 2014, pp. 215–219.

[7] D. Povey and B. Kingsbury, "Evaluation of proposed modifications to MPE for large scale discriminative training," in *Proceedings of ICASSP*, April 2007, vol. 4, pp. IV–321–IV–324.

[8] M. Gibson and T. Hain, "Hypothesis spaces for minimum bayes risk training in large vocabulary speech recognition.," in *Proceedings of INTERSPEECH*, 2006.

[9] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front end factor analysis for speaker verification," *IEEE*, 2010.

[10] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE*, vol. 15, no. 4, pp. 1435–1447, May 2007.

[11] P. Kenny, G. Boulianne, and P. Dumouchel, "Eigenvoice modeling with sparse training data," *IEEE*, vol. 13, no. 3, pp. 345–354, May 2005.

[12] G. Hinton, "A practical guide to training restricted boltzmann machines," *Momentum*, vol. 9, no. 1, pp. 926, 2010.

[13] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proceedings of ACL*, 1996, pp. 310–318.

[14] A. Stolcke, "SRILM – an extensible language modeling toolkit," in *Proceedings of ICSLP*, Denver, USA, 2002, vol. 2, pp. 901–904.

[15] T. Mikolov, S. Kombrink, L. Burget, J. Cernocky, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Proceedings of ICASSP*, 2011, pp. 5528–5531.

[16] S. Kombrink, M. Karafiat T. Mikolov, and L. Burget, "Recurrent neural network based language modeling in meeting recognition," in *Proceedings of Interspeech*, 2011, pp. 2877–2880.

[17] M. Heck, S. Mohr, C. Stüker, M. Müller, K. Kilgour, J. Gehring, QB. Nguyen, VH. Nguyen, and A. Waibel, "Segmentation of telephone speech based on speech and non-speech models," in *Speech and Computer*, M. Železný, I. Habernal, and A. Ronzhin, Eds., vol. 8113 of *Lecture Notes in Computer Science*, pp. 286–293. Springer International Publishing, 2013.

[18] H. Soltau, F. Metze, C. Fügen, and A. Waibel, "A one-pass decoder based on polymorphic linguistic context assignment," in *Proceedings of ASRU*, 2001.

[19] T. Kaukoranta, P. Fränti, and O. Nevalainen, "Iterative split-and-merge algorithm for VQ codebook generation," *Optical Engineering*, vol. 37, no. 10, pp. 2726–2732, 1998.

[20] H. Xu, D. Povey, L. Mangu, and J. Zhu, "Minimum bayes risk decoding and system combination based on a recursion for edit distance," *Computer Speech & Language*, vol. 25, no. 4, pp. 802 – 828, 2011.