ASR ERROR DETECTION AND RECOGNITION RATE ESTIMATION USING DEEP BIDIRECTIONAL RECURRENT NEURAL NETWORKS

Atsunori Ogawa and Takaaki Hori

NTT Communication Science Laboratories, NTT Corporation
{ogawa.atsunori,hori.t}@lab.ntt.co.jp

ABSTRACT

Recurrent neural networks (RNNs) have recently been applied as the classifiers for sequential labeling problems. In this paper, deep bidirectional RNNs (DBRNNs) are applied for the first time to error detection in automatic speech recognition (ASR), which is a sequential labeling problem. We investigate three types of ASR error detection tasks, i.e. confidence estimation, out-of-vocabulary word detection and error type classification. We also estimate recognition rates from the error type classification results. Experimental results show that the DBRNNs greatly outperform conditional random fields (CRFs), especially for the detection of infrequent error labels. The DBRNNs also slightly outperform the CRFs in recognition rate estimation. In addition, experiments using a reduced size of training data suggest that the DBRNNs have a better generalization ability than the CRFs owing to their word vector representation in a low-dimensional continuous space. As a result, the DBRNNs trained using only 20% of the training data show higher error detection performance than the CRFs trained using the full training data.

Index Terms— Automatic speech recognition, error detection, recognition rate estimation, deep bidirectional recurrent neural networks, generalization ability

1. INTRODUCTION

Conditional random fields (CRFs) [1] have been the most successful classification approach for addressing various types of sequential labeling problems in the fields of natural and spoken language processing. CRF is a discriminative model that calculates the probability of an output label sequence given an input feature vector sequence. By designing the observation feature functions, it can consider an input feature vector sequence across *several* preceding and/or succeeding time steps as a contextual feature vector to predict the output label of the current time. By designing the transition feature functions, it can also consider the n-gram dependency of the output labels, where n is typically two.

Recently, recurrent neural network (RNN) architectures have been successfully introduced in the language modeling for automatic speech recognition (ASR), e.g. [2–4]. A conventional n-gram language model considers only $n-1$ preceding words (where n is typically three or four) to predict the current word. In contrast, an RNN language model (RNNLM) can consider the *entire* word history by recursively propagating the activation vector through its hidden layer that has a self-loop connection. In addition, in contrast to the conventional n-gram model, which represents words with indices, the RNNLM (and also a feedforward NN-based language model) *projects* the word indices, i.e. high-dimensional one hot sparse word vectors, into a low-dimensional continuous space. With this dense word vector representation in a low-dimensional continuous space, the (R)NNLM provides a better *generalization* for unseen n-grams [5].

Following the success of RNNLMs in ASR, RNNs have recently been applied as the classifiers for several sequential labeling problems in spoken language processing. For example, in [6–8], spoken language understanding, i.e. word labeling with semantic meaning tags, is conducted using the airline travel information system corpus. And it is reported that the RNNs (both standard and modified) outperform the CRF and feedforward NN baselines. In [9], the frame-wise classification of speech recordings into three vocalization classes, i.e. laughter, filler and garbage, is conducted using the SSPNet Vocalization Corpus. And it is reported that the long short-term memory (LSTM) [10, 11] based RNNs outperform the feedforward NN baseline.

ASR error detection, e.g. confidence estimation and out-of-vocabulary (OOV) word detection, is a sequential labeling problem and CRFs have been applied as the classifiers, e.g. [12–17]. Feedforward NNs have also been applied, e.g. [14, 18–20] (in [19], an RNNLM is used as a feature extractor, but the classifier is a feedforward NN). However, to the best of our knowledge, there seems to be no study that *directly* applies RNNs as the classifiers for ASR error detection.

In this paper, *we apply RNNs as the classifiers for ASR error detection for the first time*. We employ RNNs with a *deep* [21] and *bidirectional* [22] structure, i.e. *DBRNNs* (Section 2). They have a standard tanh activation function or, alternatively, an LSTM block [10, 11] on each node in their hidden layers. We investigate three types of ASR error detection tasks, i.e. confidence estimation, OOV word detection and our proposed error type classification [15, 23, 24] (Section 3). We also *estimate recognition rates* from the error type classification results. Experimental results show that the DBRNNs greatly outperform CRFs, especially for the detection of infrequent error labels (Section 4.2). The DBRNNs also slightly outperform the CRFs in recognition rate estimation (Section 4.3). In addition, experiments using a reduced size of training data suggest that the DBRNNs have a better *generalization ability* than the CRFs owing to their word vector representation in a low-dimensional continuous space as with in RNNLMs (Section 4.4).

2. DEEP BIDIRECTIONAL RNNS

We employ the *bidirectional RNN* (*BRNN*) [22] that is shown in Fig. 1 and formulated as

$$\overrightarrow{\mathbf{h}}_t = f\left(\mathbf{W}_{\mathbf{x},\overrightarrow{\mathbf{h}}}\mathbf{x}_t + \mathbf{W}_{\overrightarrow{\mathbf{h}},\overrightarrow{\mathbf{h}}}\overrightarrow{\mathbf{h}}_{t-1} + \mathbf{b}_{\overrightarrow{\mathbf{h}}}\right), \quad (1)$$

$$\overleftarrow{\mathbf{h}}_t = f\left(\mathbf{W}_{\mathbf{x},\overleftarrow{\mathbf{h}}}\mathbf{x}_t + \mathbf{W}_{\overleftarrow{\mathbf{h}},\overleftarrow{\mathbf{h}}}\overleftarrow{\mathbf{h}}_{t+1} + \mathbf{b}_{\overleftarrow{\mathbf{h}}}\right), \quad (2)$$

$$\mathbf{y}_t = g\left(\mathbf{W}_{\overrightarrow{\mathbf{h}},\mathbf{y}}\overrightarrow{\mathbf{h}}_t + \mathbf{W}_{\overleftarrow{\mathbf{h}},\mathbf{y}}\overleftarrow{\mathbf{h}}_t + \mathbf{b}_{\mathbf{y}}\right), \quad (3)$$

where $\mathbf{x}_t$ is the input feature vector at time t, $\overrightarrow{\mathbf{h}}_t$ ($\overleftarrow{\mathbf{h}}_t$) is the activation vector at time t on the forward (backward) hidden layer, $\mathbf{W}_{\mathbf{p},\mathbf{q}}$ is the weight matrix between the two vectors $\mathbf{p}$ and $\mathbf{q}$, $\mathbf{b}_{\mathbf{r}}$ is the bias

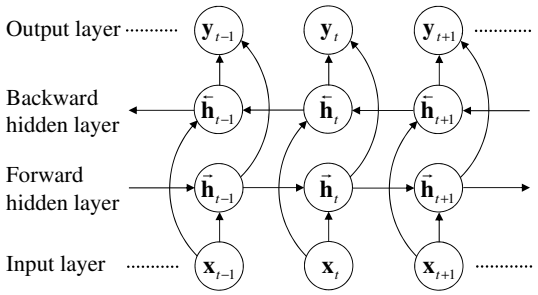

Fig. 1. Bidirectional RNN *unfolded* across time.

Table 1. Features associated with a recognized word. Features 1 to 3 are lexical or symbolic features and are each expanded to a *one hot* feature vector. The others are numerical features.

1. *Recognized word itself*	10. # of frames per phone
2. Part-of-speech	11. Acoustic log likelihood
3. Back-off behavior	12. Unigram log likelihood
4. Correct recognition prob.	13. Trigram log likelihood
5. Substitution error prob.	14. # of alternative hypotheses
6. Insertion error probability	15. # of preceding ε segments
7. Deletion error probability	16. Sum. of ε probabilities
8. Number of frames	17. Sum. of # of alter. hypos.
9. Number of phones	

term of the vector $\mathbf{r}$, $f(\cdot)$ is the activation function (e.g. tanh) on each node in the hidden layers, $g(\cdot)$ is the softmax function, and $\mathbf{y}_t$ is the posterior probability vector of the output label at time t.

In the BRNN, the forward and backward activation vectors are recursively propagated through its forward and backward hidden layers that each have a self-loop connection. Thus, *conceptually*, the BRNN can consider the context of the input feature vectors across the preceding and succeeding *full* time steps to predict the output label of the current time. This is a big advantage over the CRFs, which can consider only the context across *several* time steps. However, considering the context across the full (long) time steps is actually difficult due to the well-known *vanishing gradient problem* [25]. The standard activation function, e.g. sigmoid and tanh (we use tanh in this research), can cause this problem and, alternatively, the *long short-term memory* (*LSTM*) *block* [10, 11] has been proposed to address the problem. In this research, we use the LSTM block in addition to the tanh activation function. Using the LSTM block, we can expect to obtain better performance than with tanh, especially when the input feature vector sequence is long.

Our RNN has a *deep* structure in addition to a bidirectional structure, i.e. *DBRNN*. Several hidden layers are stacked in the forward and backward directions *individually*, i.e. there is no connection from a forward hidden layer to a backward hidden layer and vice versa (in Fig. 1, the deep structure is omitted for simplicity). The effect of the deep structure has been confirmed especially in the acoustic modeling for ASR, i.e. by using several non-linear hidden layers, we can model highly non-linear relationships between the input feature vectors and the output labels [21].

In ASR error detection, $\mathbf{x}_t$ is the input feature vector associated with the t-th recognized word in a recognition result (hypothesized word sequence). Table 1 lists the features used in this research. The first feature is the *recognized word itself* and it is expanded to a V-dimensional *one hot* word vector, where V is the vocabulary size (features 2 and 3 in Table 1 are also expanded to one hot vectors, however, their dimensionalities are much lower than V). In general, such a high-dimensional *sparse* feature vector causes the degradation of the *generalization ability* of classifiers. However, an (R)NN

Reference transcription	Recognition result	Alignment result	Label sequence 1.	2.	3.	4.
⋮	⋮	⋮	⋮	⋮	⋮	⋮
the(IV)	the	C	C	IV	C	$\overline{\text{D}}$
most(IV)	more	S	$\overline{\text{C}}$	IV	S	$\overline{\text{D}}$
	this	I	$\overline{\text{C}}$	OOV	I	$\overline{\text{D}}$
dissimilar(OOV)	similar	S	$\overline{\text{C}}$	OOV	S	$\overline{\text{D}}$
about(IV)	about	C	C	IV	C	$\overline{\text{D}}$
them(IV)		D				
is(IV)	is	C	C	IV	C	D
that(IV)	that	C	C	IV	C	$\overline{\text{D}}$
⋮	⋮	⋮	⋮	⋮	⋮	⋮

Fig. 2. Word alignment result between a recognition result and its reference transcription. (Correct) label sequences for 1. confidence estimation, 2. OOV word detection, 3. CSI classification and 4. deletion error detection. The dotted rectangle indicates the segment that is *influenced* by the utterance of an OOV word "dissimilar".

projects a sparse feature vector into a low-dimensional continuous space [2–8]. In this research, a *dense* word vector can be obtained as a column in $\mathbf{W}_{\mathbf{x},\overrightarrow{\mathbf{h}}}$ and $\mathbf{W}_{\mathbf{x},\overleftarrow{\mathbf{h}}}$ in Eqs. (1) and (2), i.e. the weight matrices between the input and the *first* forward and backward hidden layers of the DBRNN. This projection can be understood as a sort of *soft clustering* since, with this projection, different words are represented with similar vectors if they have similar ASR error trends. And we can expect that, if the projection is accurately estimated in the training, the DBRNN shows a good *generalization ability*, i.e. a good ASR error detection performance when there is a mismatch between the training and evaluation data.

The above generalization works only for words that appear in the training data. There are words that are in the vocabulary (ASR dictionary) but that do *not* appear in the training data. If such words appear in the development and/or evaluation data, *untrained* projections based on the initial random weights are applied to the words. And the ASR error detection results for the words are unreliable. To address this problem, we *zeroize* the weights for the untrained words. With this weight zeroization, ASR error detection for the untrained words can be conducted using features other than words. We found in the preliminary experiments that the effect of weight zeroization is limited but consistent. Thus, we use this technique in all the experiments described in Section 4.

3. ASR ERROR DETECTION AND RECOGNITION RATE ESTIMATION

To identify recognition errors in continuous speech recognition, a word alignment is made between a recognition result and its reference transcription using a scoring tool (e.g. NIST SCLITE scoring package [26]) based on a dynamic programming procedure. Fig. 2 shows an example of such word alignment results. In this figure, a recognized word is classified into one of three categories, i.e. correct (C), substitution error (S) and insertion error (I). A deletion error (D) is also detected. In general, ASR error detection is a task that estimates word alignment results given the recognition results *without using* their reference transcriptions.

Confidence estimation, e.g. [12–15, 18, 19], is the most basic ASR error detection task. In confidence estimation, as shown in Fig. 2, the t-th recognized word w_t in a recognition result is labeled as C or incorrect ($\overline{\text{C}}$), where $\overline{\text{C}}$ is S or I. This word labeling is performed probabilistically, i.e. $P(\text{C}|w_t) + P(\overline{\text{C}}|w_t) = 1$.

OOV word detection, e.g. [15–17, 20], can be rephrased as "the detection of misrecognized words caused by an utterance of an OOV word". Thus, as shown in Fig. 2, we have to define the segment that is *influenced* by an utterance of an OOV word. In this example, we consider that the utterance of an OOV word "dissimilar" can influence not only a directly corresponding recognized word "similar" but also *one preceding* and *one succeeding* recognized word, i.e. "this" and "about". "similar" and "this" are in the influenced segment and misrecognized. Thus, they are each labeled as an OOV word, or more precisely, "a misrecognized word caused by an utterance of an OOV word". "about" is also in the influenced segment, however, it is correctly recognized. Thus, it is labeled as an in-vocabulary (IV) word. As with the confidence estimation, $P(\mathrm{IV}|w_t) + P(\mathrm{OOV}|w_t) = 1$.

Error type classification [15, 23, 24] can be divided into two sub tasks. The first sub task is *CSI classification*. It is a simple extension of the confidence estimation and, as shown in Fig. 2, it labels the t-th recognized word w_t in a recognition result as C, S or I probabilistically, i.e. $P(\mathrm{C}|w_t)+P(\mathrm{S}|w_t)+P(\mathrm{I}|w_t) = 1$. The second sub task is *deletion error detection* (also proposed in [27]). It is a difficult task since *consecutive* deletion errors can occur at *arbitrary* inter-word positions in a recognition result. In this research, we focus on detecting the inter-word positions that have deletion errors and do not count the number of deletion errors in the inter-word position. This simplification is reasonable since, investigating the data used in the experiments described in Section 4, we found that about 80% of the deletion errors are *singletons*. We identify an inter-word position using a recognized *actual* word that *succeeds* the inter-word position. As a result, in our deletion error detection, as shown in Fig. 2, the inter-word position that *precedes* the t-th recognized actual word w_t is labeled as "the inter-word position that has *one or more* deletion errors (D)" or "that has no deletion error ($\overline{\mathrm{D}}$)" (details are provided in [23]). As with the other tasks, $P(\mathrm{D}|w_t) + P(\overline{\mathrm{D}}|w_t) = 1$. In the scoring phase, even if we can correctly detect an inter-word position that has *two or more* consecutive deletion errors, we evaluate this as the correct detection of *only one* deletion error and the other deletion errors are *not* detected.

Recognition rate estimation [15, 23, 24] can be conducted using the results of error type classification. It is an essential technique if we are to judge whether or not ASR technology is applicable to a new task at *low cost*, i.e. *without using* reference transcriptions. We obtain the CSID probabilities for the t-th recognized word w_t in a recognition result, i.e. $P(\mathrm{C}|w_t)$, $P(\mathrm{S}|w_t)$, $P(\mathrm{I}|w_t)$, $P(\mathrm{D}|w_t)$, as the results of error type classification. Using these CSID probabilities, we can estimate the numbers of CSIDs in the recognition results for an evaluation data as, e.g. $E(\#\mathrm{C}) = \sum_i \sum_t P(\mathrm{C}|w_t)$, where i is an index of a recognition result for an utterance in the evaluation data. Then, using these estimated numbers of CSIDs, i.e. $E(\#\mathrm{C})$, $E(\#\mathrm{S})$, $E(\#\mathrm{I})$, $E(\#\mathrm{D})$, we can estimate the speech recognition rates, i.e. the *percent correct* (*%Cor*) and *word accuracy* (*WAcc*) for the evaluation data as %Cor=#C/#N×100 [%] and WAcc=(#C−#I)/#N×100 [%], where #N is the estimated number of words included the evaluation data (#N=#C+#S+#D). We can say that recognition rate estimation is *easier* than the ASR error detection tasks described above since recognition rate estimation requires only the accurate estimation of the *numbers* of CSIDs while the ASR error detection tasks require the accurate *word-by-word* labeling.

4. EXPERIMENTS

We conducted experiments using the MIT lecture speech corpus [28]. We compared the DBRNNs with the CRFs as regards performance of ASR error detection and recognition rate estimation. We also conducted experiments using a reduced size of training data to investigate generalization ability of the classifiers.

Table 2. Classification accuracies [%] and F-scores [%] of 1. confidence estimation, 2. OOV word detection, 3. CSI classification and 4. deletion error detection obtained by CRF and the DBRNNs. Values in parentheses indicate ratios [%] of labels in each task. They are summed up to 100%.

1. Confidence estimation		CRF	DBRNN	DBLSTM
Classification accuracy		84.32	85.52	**85.63**
F-score	Correct (76.93)	90.16	**90.95**	90.94
	Incorrect (23.07)	66.77	69.23	**69.24**
2. OOV word detection		CRF	DBRNN	DBLSTM
Classification accuracy		94.29	**94.61**	94.60
F-score	In-vocab. (94.07)	97.19	**97.21**	97.20
	OOV (5.93)	43.72	47.02	**47.54**
3. CSI classification		CRF	DBRNN	DBLSTM
Classification accuracy		82.13	**83.33**	83.25
F-score	Correct (76.93)	90.27	90.96	**90.97**
	Substitution (18.30)	59.58	61.87	**61.89**
	Insertion (4.77)	39.22	**43.96**	43.93
4. Deletion error detection		CRF	DBRNN	DBLSTM
Classification accuracy		96.25	**96.45**	**96.45**
F-score	No deletion (96.44)	98.20	**98.22**	**98.22**
	Deletion (3.56)	30.01	**34.30**	33.38

4.1. Experimental Settings

The ASR settings were basically the same as those described in [15, 23, 24]. A discriminatively trained GMM-HMM-based acoustic model [29] and a word trigram with a 16.5k vocabulary size were used in the SOLON decoder [30].

The classifier training data consisted of 215 hours (238 lectures) of speech (114k utterances and 2M words). The development data consisted of 2.3 hours (2 lectures) of speech (3k utterances and 22k words). And the evaluation data consisted of 7 hours (8 lectures) of speech (6.5k utterances and 72k words). ASR was conducted on these data and the features listed in Table 1 were extracted. The labels for each of the ASR error detection tasks shown in Fig. 2 were also obtained with SCLITE [26] using the reference transcriptions.

Using the pairs of features and labels of the training data, we trained the DBRNNs for each of the ASR error detection tasks. We used RNNLIB [31–33] for the training. We modified this tool to make it possible to efficiently process the sparse input feature vectors. The weight matrices and bias terms in the DBRNNs were randomly initialized and then updated with the stochastic gradient descent and the backpropagation through time, which was truncated at the beginning and end of an utterance, based on the minimization of cross-entropy error. The training was early terminated by tracking the classification accuracies for the development data. From the results of the preliminary experiments, we set the structure of the DBRNNs so that there were three hidden layers and 20 nodes at each hidden layer. Thus, a one hot sparse word vector 16.5k in size was projected into a $40(= 20(\mathrm{forward}) + 20(\mathrm{backward}))$-dimensional continuous space dense word vector.

We also trained the CRFs for each of the ASR error detection tasks using CRF++ [34]. The observation feature functions were designed to take account of the input feature vectors across the preceding and succeeding two words, i.e. the contextual input feature vector for the five words. The bigram dependency of the output labels was also considered by designing the transition feature functions.

In addition to the CRFs and DBRNNs, we also trained and evaluated deep *feedforward* NNs and deep *unidirectional* RNNs. However, their performance fell between those of the CRFs and

DBRNNs. Thus, we do not describe their results to save space. Hereafter, the CRFs are denoted "CRF". The DBRNNs based on the standard tanh activation function are denoted "DBRNN" and those based on the LSTM block are denoted "DBLSTM".

4.2. ASR Error Detection Results

Table 2 shows the results of ASR error detection. In addition to the classification accuracies, the F-scores for detecting each type of label are shown. This is because the frequencies of each type of label are highly *unbalanced*. We can confirm that the DBRNNs greatly outperform CRF, especially for the detection of the infrequent error labels, e.g. OOV, insertion error and deletion error. However, we cannot confirm the superiority of DBLSTM over DBRNN. We think that one reason for this is attributable to the relatively short length (about 11 words on average) of the recognition results for the evaluation data. Thus, hereafter, we do not describe the results obtained by DBLSTM.

4.3. Recognition Rate Estimation Results

Table 3 and Fig. 3 show the estimation results of the recognition rates. As described in Section 3, recognition rate estimation is easier than ASR error detection. And both CRF and DBRNN show very high estimation performance. However, the details confirm that DBRNN slightly outperforms CRF.

4.4. Experimental Results Using Reduced Size of Training Data

To investigate the generalization ability of the classifiers, we reduced the size of the training data from 100% to 10% in 10% steps. As the size of the training data was reduced, the mismatch between the training and evaluation data became large.

Figure 4 shows the experimental results for the CSI classification accuracies and the F-scores of insertion error detection (blue and red curves). When the training data size was reduced from 20% to 10%, the performance of both CRF and DBRNN quickly degraded. This indicates that 10% of the data is insufficient to train the classifiers accurately. Thus, by comparing CRF and DBRNN when reducing the data size from 100% to 20%, we can confirm that the performance degradation of DBRNN is smaller than that of CRF (in the other measures, the trends are also similar to these two curves). There results suggest that DBRNN has a better generalization ability than CRF. As a result, DBRNN trained using only 20% of the data performs better than CRF trained using the full data.

One idea for further increasing the generalization ability of the classifiers is to *not* use the one hot sparse word vectors as in [19]. The experimental results for such a case are also shown in Fig. 4 (aqua and pink curves). In this case, CRF shows a better generalization ability than DBRNN since the performance degradation of CRF is smaller than that of DBRNN when the training data size is reduced. However, DBRNN maintains its high *absolute* performance. For example, DBRNN *without* the one hot word vectors and trained using 30-40% of the data performs better than CRF with the one hot word vectors and trained using the full data.

5. CONCLUSION AND FUTURE WORK

We applied DBRNNs as the classifiers for ASR error detection and recognition rate estimation for the first time. Experimental results showed that the DBRNNs substantially outperformed the CRFs. In addition, the DBRNNs showed a better generalization ability than the CRFs. Future work will include improving the performance of the LSTM based DBRNNs as investigated in [35]. Techniques for dealing with unbalanced label frequencies will also be required for further improvement in the detection of infrequent error labels as indicated in [9].

Table 3. Number of NCSIDs, percent correct rates [%] and word accuracies [%] for the entire evaluation data (6482 utterances) estimated by CRF and DBRNN. True values are calculated by SCLITE [26] using reference transcriptions.

	#N	#C	#S	#I	#D	%Cor	WAcc
True	72283	55613	13231	3450	3439	76.94	72.16
CRF	71925	54825	13966	**3504**	3134	76.23	71.35
DBRNN	**72314**	**55735**	**13312**	3246	**3267**	**77.07**	**72.58**

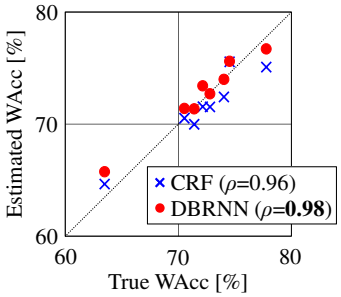

Fig. 3. Correlations between lecture-level (eight lectures) true word accuracies and those estimated by CRF or DBRNN.

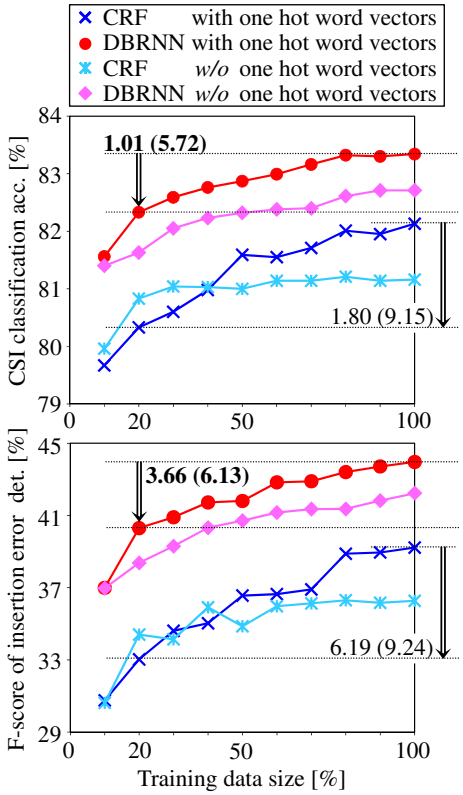

Fig. 4. (Top) CSI classification accuracies and (bottom) F-scores of insertion error detection as a function of training data size obtained by CRF and DBRNN with or *without* one hot sparse word vectors. Values close to the down arrows "⇓" indicate absolute (relative) performance degradation rates [%] of the classifiers using the one hot sparse word vectors when reducing the training data size from 100% to 20%.

6. REFERENCES

[1] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: probabilistic models for segmenting and labeling sequence data," in *Proc. International Conference on Machine Learning (ICML)*, 2001, pp. 282–289.

[2] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. Interspeech*, 2010, pp. 1045–1048.

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

[4] T. Hori, Y. Kubo, and A. Nakamura, "Real-time one-pass decoding with recurrent neural network language model for speech recognition," in *Proc. ICASSP*, 2014, pp. 6364–6368.

[5] H. Schwenk, "Continuous space language model," *Computer Speech and Language*, vol. 21, no. 3, pp. 492–518, July 2007.

[6] K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding," in *Proc. Interspeech*, 2013, pp. 2524–2528.

[7] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent neural network architectures and learning methods for spoken language understanding," in *Proc. Interspeech*, 2013, pp. 3771–3775.

[8] K. Yao, B. Peng, G. Zweig, D. Yu, X. Li, and F. Gao, "Recurrent conditional random field for language understanding," in *Proc. ICASSP*, 2014, pp. 4105–4109.

[9] R. Brueckner and B. Schuller, "Socal signal classification using deep BLSTM recurrent neural networks," in *Proc. ICASSP*, 2014, pp. 4856–4860.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, November 1997.

[11] F.A. Gers, N.N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," *Journal of Machine Learning Research*, vol. 3, pp. 115–143, 2002.

[12] J. Fayolle, F. Moreau, C. Raymond, G. Gravier, and P. Gros, "CRF-based combination of contextual features to improve a posteriori word-level confidence measures," in *Proc. Interspeech*, 2010, pp. 1942–1945.

[13] M.S. Seigel and P.C. Woodland, "Combining information sources for confidence estimation with CRF models," in *Proc. Interspeech*, 2011, pp. 905–908.

[14] D. Yu, J. Li, and L. Deng, "Calibration of confidence measures in speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 8, pp. 2461–2473, November 2011.

[15] A. Ogawa, T. Hori, and A. Nakamura, "Error type classification and word accuracy estimation using alignment features from word confusion network," in *Proc. ICASSP*, 2012, pp. 4925–4928.

[16] C. Parada, M. Dredze, D. Filimonov, and F. Jelinek, "Contextual information improves OOV detection in speech," in *Proc. HLT-NAACL*, 2010, pp. 216–224.

[17] A. Marin, T. Kwiatkowski, M. Ostendorf, and L. Zettlemoyer, "Using syntactic and confusion network structure for out-of-vocabulary word detection," in *Proc. IEEE Workshop on Spoken Language Technology (SLT)*, 2012, pp. 159–118.

[18] P.-S. Huang, K. Kumar, C. Liu, Y. Gong, and L. Deng, "Predicting speech recognition confidence using deep learning with word identity and score features," in *Proc. ICASSP*, 2013, pp. 7413–7417.

[19] Y.-C. Tam, Y. Lei, J. Zheng, and W. Wang, "ASR error detection using recurrent neural network language model and complementary ASR," in *Proc. ICASSP*, 2014, pp. 2331–2335.

[20] S. Kombrink, L. Burget, P. Matějka, M. Karafiát, and H. Hermansky, "Posterior-based out of vocabulary word detection in telephone speech," in *Proc. Interspeech*, 2009, pp. 80–83.

[21] G. Hinton, L. Deng, D. Yu, G.E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, November 2012.

[22] M. Schuster and K.K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, November 1997.

[23] A. Ogawa, T. Hori, and A. Nakamura, "Recognition rate estimation based on word alignment network and discriminative error type classification," in *Proc. IEEE Workshop on Spoken Language Technology (SLT)*, 2012, pp. 113–118.

[24] A. Ogawa, T. Hori, and A. Nakamura, "Discriminative recognition rate estimation for n-best list and its application to n-best rescoring," in *Proc. ICASSP*, 2013, pp. 6832–6836.

[25] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," in *A Field Guide to Dynamical Recurrent Networks*, S.C. Kremer and J.F. Kolen, Eds., pp. 237–243. Wiley-IEEE Press, 2001.

[26] "NIST SCLITE Scoring Package Version 1.5," http://www.icsi.berkeley.edu/Speech/docs/sctk-1.2/sclite.htm.

[27] M.S. Seigel and P.C. Woodland, "Detecting deletions in ASR output," in *Proc. ICASSP*, 2014, pp. 2321–2325.

[28] J. Glass, T.J. Hazen, S. Cyphers, I. Malioutov, D. Huynh, and R. Barzilay, "Recent progress in the MIT spoken lecture processing project," in *Proc. Interspeech*, 2007, pp. 2553–2556.

[29] E. McDermott, S. Watanabe, and A. Nakamura, "Discriminative training based on an integrated view of MPE and MMI in margin and error space," in *Proc. ICASSP*, 2010, pp. 4894–4897.

[30] T. Hori, C. Hori, Y. Minami, and A. Nakamura, "Efficient WFST-based one-pass decoding with on-the-fly hypothesis rescoring in extremely large vocabulary continuous speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1352–1365, May 2007.

[31] "RNNLIB," http://sourceforge.net/projects/rnnl/.

[32] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, July-August 2005.

[33] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. ICASSP*, 2013, pp. 6645–6649.

[34] "CRF++0.58," https://code.google.com/p/crfpp/.

[35] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proc. Interspeech*, 2012.