RESCORING N-BEST SPEECH RECOGNITION LIST BASED ON ONE-ON-ONE HYPOTHESIS COMPARISON USING ENCODER-CLASSIFIER MODEL

Atsunori Ogawa, Marc Delcroix, Shigeki Karita, and Tomohiro Nakatani

NTT Communication Science Laboratories, NTT Corporation, Kyoto, Japan

ABSTRACT

This paper proposes a new model for accurately rescoring (reranking) N-best speech recognition hypothesis lists. The model is based on state-of-the-art neural networks (NNs) and provides the minimum necessary functionality to perform N-best rescoring, i.e. one-on-one hypothesis comparison on a given N-best list in terms of word error rates (WERs). The model is composed of a long short-term memory (LSTM)-based encoder network followed by a fully-connected feedforward NN-based binary-class classifier network. Given the feature vector sequences of two hypotheses to be compared, this encoder-classifier (EC) model encodes these features and outputs binary-class probabilities that indicate which hypothesis has the lower WER. Then, depending on the output, the ranks of these hypotheses can be swapped. By repeating this one-on-one hypothesis comparison, a reranked N-best list can be obtained. In N-best rescoring experiments using a large scale speech corpus, the proposed EC model steadily outperforms an LSTM-based language model (LSTMLM), which is a strong and widely-used competitor. In addition, by incorporating the LSTMLM scores as an additional feature vector dimension, the N-best rescoring performance of the EC model is further improved. The improved EC model achieves a 10% relative WER reduction from the LSTMLM baseline.

Index Terms— Speech recognition, N-best list, N-best rescoring, one-on-one hypothesis comparison, encoder-classifier model

1. INTRODUCTION

Recently, great progress has been made on automatic speech recognition (ASR) technology based on the introduction of state-of-the-art neural networks (NNs) [1, 2], and various types of ASR-based applications, including voice search services and spoken dialogue systems, have been actively developed. Despite this great progress, in some situations such as performing ASR in noisy environments and/or performing ASR for casual-style speech, ASR accuracy remains at an unsatisfactory level [3, 4].

Some tasks that require high ASR accuracy employ multiple speech recognition hypotheses (word sequences), which are represented in the form of, e.g. the N-best list, word lattice, and word confusion network [5]. This is because 1-best speech recognition hypotheses may contain many errors in the severe situations described above, but hypotheses that have significantly lower word error rates (WERs) than the 1-best hypotheses can be found in multiple hypotheses if they are rescored (reranked). For example, in the CHiME-4 noisy speech recognition challenge [6], the top scoring systems performed N-best rescoring [7, 8] or lattice rescoring [9, 10] using recurrent NN language models (RNNLMs) at the final step of ASR. N-best lists have also been exploited in spoken dialogue systems [11, 12].

In this paper, we focus on the *rescoring of N-best speech recognition hypothesis lists*. Currently the most widely-used models for N-best rescoring are RNNLMs [13, 14], and in particular, long short-term memory (LSTM)-based RNNLMs (LSTMLMs) [7, 8, 15]. An LSTMLM has much better word prediction ability than a conventional n-gram LM. It can be trained using the same text corpus (i.e. reference (correct) transcriptions) used for training an n-gram LM. In other words, except in a few studies [16, 17], an LSTMLM cannot be trained while taking ASR errors into account. An LSTMLM provides scores for each of the hypotheses in a given N-best list. These scores are interpolated with the ASR scores attached to each of the hypotheses and these interpolated scores are used to rerank the hypotheses in the list. Note that, even though an LSTMLM performs well in N-best rescoring, it was originally developed to predict the next word (the vocabulary size is usually very large) and was not developed to undertake N-best rescoring.

In contrast to the LSTMLMs, discriminative language models (DLMs) [18–21] were originally developed for N-best rescoring. A DLM can be trained taking ASR errors into account using a training data set other than that used for training the n-gram LM. A DLM is based on a log-linear model that uses n-gram counts as features. Its model parameters (i.e. the weights of the features) are optimized based on the minimization of a loss function, using N-best lists provided by an ASR system for the training data set. The loss function is designed so that the trained DLM can perform hypotheses reranking for a given N-best list. With the loss function, two hypotheses are compared in terms of their absolute WERs for all (or selected) pairs of hypotheses in all the N-best lists of the training data set. As with the LSTMLM, a trained DLM provides scores for each of the hypotheses in a given N-best list, and these scores are interpolated with the ASR scores to perform hypothesis reranking of the list. The design of the loss function is sophisticated. It simultaneously takes into account all (or many) pairs of hypotheses to be compared. However, as described below, we can develop a model based on a simpler design for N-best rescoring.

Let us consider the minimum functionality needed to perform N-best rescoring. For a given N-best list, if we can judge that the WER of the vth hypothesis is lower than that of the uth hypothesis ($u < v$), we can swap the ranks of these hypotheses. As with traditional sorting algorithms [22], by repeating this one-on-one hypothesis comparison for pairs of hypotheses in the list and, if necessary, by swapping the ranks of the hypotheses, we can obtain a reranked N-best list. All we need to do is to judge which hypothesis has the lower WER. In this paper, we refer to this judgment as *one-on-one hypothesis comparison* (duel), and this is the minimum necessary functionality to perform N-best rescoring.

We propose a new N-best rescoring model based on state-of-the-art NNs (Section 2). In contrast to the LSTMLMs and DLMs, the model focuses only on providing the minimum necessary functionality to perform N-best rescoring, i.e. *one-on-one hypothesis comparison* on a given N-best list. The model is composed of an LSTM-based *encoder* network [23–25] followed by a fully-connected feedforward (FCFF) NN-based binary-class *classifier* network. We also propose efficient procedures for training the proposed *EC model* (Section 2.2) and for performing N-best rescoring using the trained EC model (Section 2.3). In experiments using a large scale speech corpus (Section 3), the EC model steadily outperforms an LSTMLM,

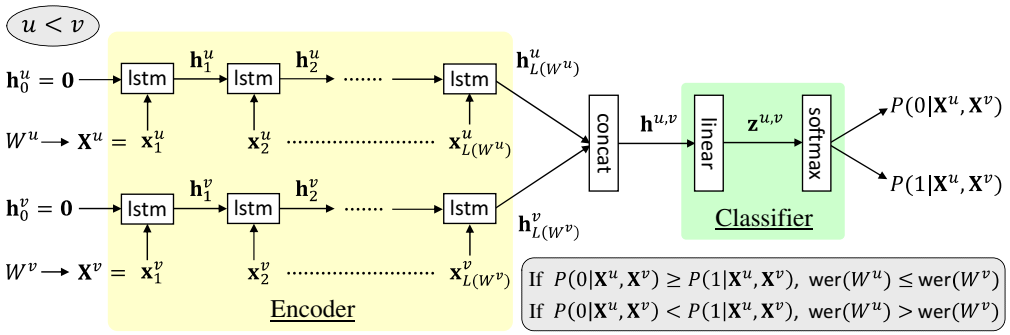

Fig. 1. Proposed encoder-classifier (EC) model that performs one-on-one hypothesis comparison for N-best rescoring.

which is a widely-used strong competitor. In addition, by using the LSTMLM scores as an additional feature, the N-best rescoring performance of the EC model can be further improved.

2. ENCODER-CLASSIFIER MODEL

We propose a new N-best rescoring model, called an *encoder-classifier* (*EC*) *model*. We also propose procedures for its efficient training and evaluation.

2.1. Structure of EC Model

The propsed EC model is based on state-of-the-art NNs and the focus is on providing the minimum necessary functionality to perform N-best rescoring. Given an N-best list, the EC model performs a one-on-one hypothesis comparison in terms of the WERs of two hypotheses to be compared.

Let $W^u = w_1^u, w_2^u, \cdots, w_{L(W^u)}^u$ be the uth hypothesis (word sequence) of length (number of words) $L(W^u)$ in a given N-best list. $\mathbf{A}^u = \mathbf{a}_1^u, \mathbf{a}_2^u, \cdots, \mathbf{a}_{L(W^u)}^u$ is the auxiliary feature vector sequence that corresponds to W^u. The auxiliary feature vector $\mathbf{a}_i^u$ for the word w_i^u is, for example, composed of an acoustic score (log likelihood) and a linguistic score (log probability) provided by an ASR system to the word w_i^u. $\mathbf{X}^u = \mathbf{x}_1^u, \mathbf{x}_2^u, \cdots, \mathbf{x}_{L(W^u)}^u$ is the feature vector sequence that corresponds to W^u. The feature vector $\mathbf{x}_i^u$ of the word w_i^u is obtained as $\mathbf{x}_i^u = \mathtt{concat}(\mathtt{embed}(w_i^u), \mathbf{a}_i^u)$, where $\mathtt{embed}(\cdot)$ denotes the NN-based word embedding operation and $\mathtt{concat}(\cdot)$ denotes the vector concatenation operation.

Figure 1 shows the proposed EC model that performs a one-on-one hypothesis comparison for N-best rescoring. Given the feature vector sequences of the uth hypothesis W^u and the vth hypothesis W^v ($u < v$), i.e. $\mathbf{X}^u$ and $\mathbf{X}^v$, the proposed model is assumed to output the probabilities of binary class symbols, i.e. $P(y|\mathbf{X}^u, \mathbf{X}^v)$ for $y = \{0, 1\}$, so that they satisfy,

$$\begin{cases} P(0|\mathbf{X}^u, \mathbf{X}^v) \geq P(1|\mathbf{X}^u, \mathbf{X}^v) & \text{if } \mathtt{wer}(W^u) \leq \mathtt{wer}(W^v), \\ P(0|\mathbf{X}^u, \mathbf{X}^v) < P(1|\mathbf{X}^u, \mathbf{X}^v) & \text{otherwise}, \end{cases} \tag{1}$$

where $\mathtt{wer}(\cdot)$ is a function that returns the WER of a given hypothesis and $\sum_y P(y|\mathbf{X}^u, \mathbf{X}^v) = 1$. If the upper inequality is satisfied, the model estimates that the WER of W^u is not greater than that of W^v, and their ranks remain unchanged. Conversely, if the lower inequality is satisfied, the model estimates that the WER of W^u is higher than that of W^v, and we swap their ranks.

The lengths of W^u and W^v, i.e. $L(W^u)$ and $L(W^v)$, may differ. To deal with hypotheses with different lengths, we employ a unidirectional LSTM unit-based *encoder* network [23–25] in the proposed model. By using an encoder network, these two hypotheses can be represented with fixed-length encoded hidden state vectors, and the hypotheses can be fairly compared by using their encoded vectors. Given the feature vector $\mathbf{x}_i^u$ of the current word w_i^u and the previous hidden state vector $\mathbf{h}_{i-1}^u$, a one-layer LSTM unit provides the current hidden state vector $\mathbf{h}_i^u$ as follows,

$$\mathbf{h}_i^u = \mathtt{lstm}(\mathbf{x}_i^u, \mathbf{h}_{i-1}^u), \tag{2}$$

where $\mathtt{lstm}(\cdot)$ denotes the operation of a one-layer LSTM unit and $\mathbf{h}_0^u = \mathbf{0}$. $\mathbf{h}_i^u$ encodes the feature vector sequence $\mathbf{x}_1^u, \mathbf{x}_2^u, \cdots, \mathbf{x}_i^u$ of the word sequence $w_1^u, w_2^u, \cdots, w_i^u$. By repeating this operation for each feature vector $\mathbf{x}_i^u$ in $\mathbf{X}^u$, we can obtain the hidden state vector $\mathbf{h}_{L(W^u)}^u$ that encodes $\mathbf{X}^u$. By also applying this operation to $\mathbf{X}^v$, we can obtain the encoded hidden state vector $\mathbf{h}_{L(W^v)}^v$ of $\mathbf{X}^v$ (parameters are shared between the LSTMs for encoding $\mathbf{X}^u$ and $\mathbf{X}^v$). Then, we concatenate $\mathbf{h}_{L(W^u)}^u$ and $\mathbf{h}_{L(W^v)}^v$ to obtain the concatenated encoded hidden state vector $\mathbf{h}^{u,v}$ as follows,

$$\mathbf{h}^{u,v} = \mathtt{concat}(\mathbf{h}_{L(W^u)}^u, \mathbf{h}_{L(W^v)}^v). \tag{3}$$

The *classifier* network follows the encoder network. The classifier network is composed of a one-layer FCFF linear layer followed by the softmax activation function. The concatenated encoded hidden state vector $\mathbf{h}^{u,v}$ provided by the encoder network is input to the classifier network and, finally, we obtain the binary class probabilities, i.e. $P(y|\mathbf{X}^u, \mathbf{X}^v)$ for $y = \{0, 1\}$, as follows,

$$\begin{aligned} \mathbf{z}^{u,v} &= \mathtt{linear}(\mathbf{h}^{u,v}), \\ P(y|\mathbf{X}^u, \mathbf{X}^v) &= \mathtt{softmax}(\mathbf{z}^{u,v})_y, \end{aligned} \tag{4}$$

where $\mathtt{linear}(\cdot)$ and $\mathtt{softmax}(\cdot)$ denote the operations of a one-layer FCFF linear layer and the softmax activation function, respectively.

In the above explanation, we used a one-layer unidirectional LSTM unit for the encoder network and a one-layer FCFF linear layer for the classifier network. We can also use multiple layers instead and, as regards the encoder network, we can also use a bidirectional LSTM unit. Furthermore, we can exploit the abilities of other N-best rescoring models in the EC model by adding their scoring results as additional dimensions of the auxiliary feature vectors.

2.2. Efficient Training of EC Model

We train the proposed EC model by feeding it with pairs of hypotheses and their corresponding binary teacher labels (i.e. $y = \{0, 1\}$). From an N-best list for an input utterance, we can extract ${}_N\mathrm{C}_2$ pairs of hypotheses. Typically, N is set at 100 to 1000, and thus the number of pairs in the list is large. In addition, a training data set usually contains more than tens of thousands of utterances. Thus, the total

number of hypothesis pairs in the training data set is huge and, in reality, it is impossible to use all the pairs for model training.

The main purpose of N-best rescoring is to find the *oracle* (i.e. the lowest WER) hypothesis from a given N-best list in the evaluation phase. In other words, as long as we can find the oracle hypothesis from the list, we do not always need to be concerned about the ranks and/or the order of the other remaining hypotheses. We have developed an efficient EC model training procedure that focuses on achieving this purpose.

First, we make a set of the selected hypotheses from an N-best list to efficiently perform one-on-one hypothesis comparisons during the EC model training. All the comparisons are performed between the oracle hypothesis and one of the other selected hypotheses. In all the comparisons (duel), the oracle hypothesis has to defeat the competitor hypothesis (i.e. the EC model has to estimate that the oracle hypothesis has a lower WER than the competitor hypothesis). To guarantee the minimum variety of the hypothesis comparisons, we preferentially select the following four hypotheses as the competitors: 1) the 1-best (i.e. the highest ASR score) hypothesis, 2) the second lowest WER hypothesis, 3) the lowest rank (i.e. the lowest ASR score) hypothesis, and 4) the highest WER hypothesis. Hypotheses 1) and 2) are selected as strong competitors that are difficult to beat. Conversely, hypotheses 3) and 4) are selected as the weak competitors that are easy to beat. To increase the variety of the hypothesis comparisons, in addition to the above four hypotheses, we also include in the set hypotheses selected from the N-best list with an equal rank interval.

Let us assume that, as a result of the hypothesis selection described above, M hypotheses (including the oracle hypothesis) are selected in the set. From this set, we can extract $M-1$ pairs between the oracle and competitor hypotheses. We found in the preliminary experiments that M can be set at a few dozen, i.e. $M \ll {}_N\mathrm{C}_2$, and the efficiency of the EC model training can be greatly increased.

Second, we train the EC model by feeding it with the above obtained $M-1$ hypothesis pairs and their corresponding binary class teacher labels. Figure 2 (top) shows the core part of the training procedure. We assign the oracle hypothesis to the uth hypothesis W^u and the competitor hypothesis to the vth hypothesis W^v ($u < v$). The features of this hypothesis pair, i.e. $(\mathbf{X}^u, \mathbf{X}^v)$, and the teacher label $y = 0$ that indicates $\mathtt{wer}(W^u) \leq \mathtt{wer}(W^v)$ are fed to the EC model and its parameters are updated. We perform this core procedure again by swapping the hypotheses and changing the teacher label binary value (i.e. we double the training data). We repeat this procedure for each oracle and competitor hypothesis pair in all the selected hypothesis sets. With this procedure, the EC model is trained so that it can distinguish the oracle hypothesis from the other hypotheses in a given N-best list in the evaluation phase.

2.3. Finding Oracle Hypothesis Using EC model

Using the trained EC model, in the evaluation phase, we find the oracle hypothesis from a given N-best list as shown in Fig. 2 (bottom). As with the first step of the bubble sort [22], we repeat a one-on-one hypothesis comparison (duel) from the top to the bottom of the list. Then, we select the surviving hypothesis as the oracle hypothesis.

Note that, as with the N-best rescoring procedure using the conventional N-best rescoring models, also in hypothesis comparisons using the EC model, we can use ASR scores attached to each of the hypotheses. In this case, we use the binary class symbol probabilities provided by the EC model as the additional scores (log probabilities) and obtain their weighted sum with the corresponding ASR scores (log likelihoods). We can obtain the scores of W^u and W^v ($u < v$),

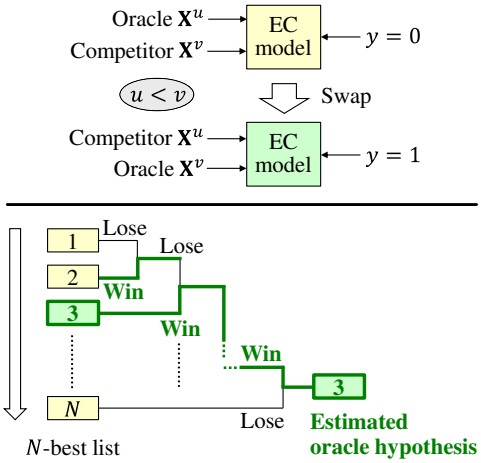

Fig. 2. (Top) Core part of training procedure. (Bottom) One-on-one hypothesis comparison (duel) to find oracle hypothesis.

i.e. $s(W^u)$ and $s(W^v)$, used for hypothesis comparison as follows,

$$\begin{cases} s(W^u) = (1-\lambda)\log p(W^u|\mathbf{O}) + \lambda \log P(0|\mathbf{X}^u, \mathbf{X}^v), \\ s(W^v) = (1-\lambda)\log p(W^v|\mathbf{O}) + \lambda \log P(1|\mathbf{X}^u, \mathbf{X}^v), \end{cases} \quad (5)$$

where $\mathbf{O}$ is the acoustic feature vector sequence of an input utterance, $\log p(W^r|\mathbf{O})$ is the ASR score provided to the rth hypothesis W^r, and λ is the weight for the EC model scores ($0 \leq \lambda \leq 1$).

3. EXPERIMENTS

We conducted experiments using the corpus of spontaneous Japanese (CSJ) [26], which is a large scale lecture speech corpus. We compared the proposed EC model with an LSTMLM [7, 8, 15] in terms of the WER of the 1-best hypotheses obtained by performing N-best rescoring. We also evaluated an EC model that exploits scores provided by the LSTMLM as an additional auxiliary feature vector dimension. In the following sections, the training and evaluation of the NN-based models were both performed using Chainer [27].

3.1. Experimental Settings

The data set for training ASR models consisted of 250 hours of speech, 198k utterances, and 3.4M words. Using this data, we trained a convolutional NN (CNN)-based acoustic model [28], a trigram LM, and an LSTMLM. The vocabulary size of these LMs was set at 31k (the words that appear only one time in the training data were mapped to the unknown word). The structure of the LSTMLM is shown in Table 1. Its parameters were randomly initialized and then iteratively updated with a stochastic gradient descent (SGD) optimizer and the backpropagation through time algorithm, based on the minimization of the cross-entropy loss. Details of the LSTMLM training recipe can be found on the Chainer homepage [27] (we found that this recipe also worked very well in our experiments).

Table 2 shows details of the EC model training, development, and evaluation data sets. We performed ASR on these data sets with a weighted finite state transducer (WFST)-based one-pass speech recognizer [29] using the CNN acoustic model and the trigram LM described above, and we obtained a 200-best hypothesis list for each of the utterances in these data sets. At the same time, the 17-dimensional ASR-based auxiliary feature (e.g. the acoustic and linguistic scores) vector was extracted word-by-word for each of the

Table 1. Structures of the LSTMLM and EC model. In both models, # nodes are the same for the word embedding layers, unidirectional LSTM units, and FCFF linear layers.

	LSTMLM	EC
# nodes	650	100
# LSTM layers	2	1
# FCFF layers	1	1
Softmax output size	31k	2
# total parameters	47M	2.9M

hypotheses. These features are detailed in [30]. To efficiently train the EC model, as described in Section 2.2, we performed hypothesis selection for the training data. We selected 20 (=M) hypotheses (including the oracle hypothesis) for each of the 200 (=N)-best lists in the training data. The selected hypotheses were used for the EC model training with the procedure described in Section 2.2.

The structure of the EC model is shown in Table 1. Note that its total number of parameters (2.9M) is very small compared with that of the LSTMLM (47M). The 100-dimensional embedded word vectors are concatenated with the 17-dimensional auxiliary feature vectors and fed to the EC model. We randomly initialized all the parameters in the EC model and then iteratively updated them with an Adam optimizer ($\alpha = 0.0001$) [31] and the backpropagation through time algorithm, which was truncated at the beginning of the hypotheses (the batch size was set at ten utterances), based on the minimization of the cross-entropy loss. We iterated the training for ten epochs and selected the model that provided the lowest loss for the development data as the model used for the evaluation.

As described in Section 2.1, we also trained an EC model that exploited scores provided by the LSTMLM as the 18th dimension of the auxiliary feature vectors. The LSTMLM provided a score (log probability) word-by-word for each of the hypotheses in all the data sets. Then, an EC model, which had the same structure as the above described EC model (Table 1), was trained using feature vectors consisting of the 100-dimensional embedded word vectors and the 18-dimensional extended auxiliary feature vectors.

In the evaluation phase, we performed N-best rescoring using the LSTMLM and the two EC models described above by taking the weighted sum of their scores and the ASR scores (see Section 2.3 for the N-best rescoring procedure using the EC models). The weight λ for these models was changed from 0 to 1 in 0.05 steps. When λ was set at 1, these models performed N-best rescoring alone (i.e. they did not use the ASR scores). For each model, the λ value that provided the best N-best rescoring performance (i.e. the lowest 1-best hypothesis WER) for the development data was selected as the optimal value, and this value was used for N-best rescoring on the evaluation data.

3.2. Experimental Results

Table 3 shows N-best rescoring results obtained with the three models for the development and evaluation data. We can confirm that the LSTMLM steadily reduces the WERs from the ASR baselines and the EC model provides additional WER reductions from the good LSTMLM baselines. We can also confirm that the EC model trained with the LSTMLM score provides further large WER reductions. These results indicate that the LSTMLM and EC model have complementary abilities, and the EC model can efficiently incorporate the ability of the LSTMLM. It achieves 10% relative WER reductions from the LSTMLM baselines.

Figure 3 shows the WERs for the development data obtained with the three models as a function of the weight λ for these models. We can confirm that there is an optimal value range of λ for the LSTMLM (at around 0.8) and it needs to be used with the ASR scores since its WER when $\lambda = 1$ becomes much worse. Conversely, the optimal values of λ for the EC models range very widely (around 0.4 to 1) and they need not be used with the ASR scores since they show the best (or nearly the best) WERs when $\lambda = 1$. This robustness against the weight λ is also an advantage of the EC model over the LSTMLM.

Table 2. Details of the EC model training (ECTr), development (Dvlp), and evaluation (Eval) data sets. "OOV" denotes the out-of-vocabulary rate [%]. "3g" and "LSTMLM" denote the perplexities obtained with the trigram LM and LSTMLM. #hyps of ECTr is the number after performing hypothesis selection.

Data	hours	#utts	#words	#hyps	OOV	3g	LSTM
ECTr	271	215k	3.5M	3.6M	2.18	—	—
Dvlp	6.0	4822	78k	670k	2.53	79.3	42.5
Eval	5.7	3163	70k	463k	1.87	77.7	48.3

Table 3. N-best rescoring results in WER [%] for the development and evaluation data obtained with the LSTMLM, EC model, and EC model trained with the LSTMLM score. WERs of the ASR (trigram) baseline and oracle hypotheses are also shown.

Model	Dvlp	Eval
ASR (trigram) baseline	18.1	14.8
LSTMLM baseline	17.3	14.2
EC model	16.6	13.8
EC with LSTMLM score	**15.5**	**12.8**
Oracle	11.1	9.3

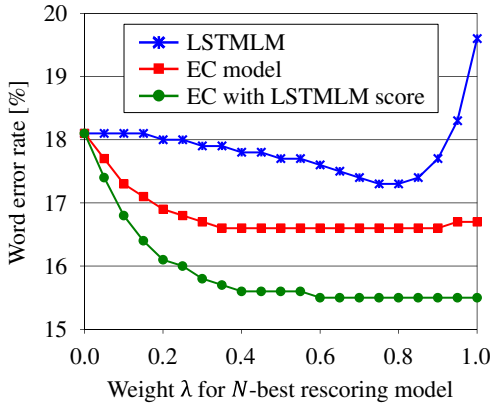

Fig. 3. WERs for the development data obtained with the LSTMLM, EC model, and EC model trained with the LSTMLM score as a function of the weight λ for these N-best rescoring models.

4. CONCLUSION AND FUTURE WORK

We have proposed a new N-best rescoring model. The proposed encoder-classifier (EC) model performs a one-on-one hypothesis comparison on a given N-best list. We have confirmed its superior N-best rescoring performance experimentally. This is a general framework, and it can be applied in other research fields that use the N-best list form of hypotheses, e.g. machine translation [32, 33].

Future work will include comparison with the DLMs [18–21] and the discriminatively trained LSTMLMs [16, 17]. We also plan to improve the performance of the proposed EC model with (1) the use of multi-layered networks, (2) the use of bidirectional LSTM units, (3) an increase in the model size, and (4) the introduction of an attention mechanism [23–25].

5. REFERENCES

[1] 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, Nov. 2012.

[2] D. Yu and L. Deng, *Automatic speech recognition: A deep learning approach*, Springer-Verlag London, 2015.

[3] J. Li, L. Deng, Y. Gong, and R. Haeb-Umbach, "An overview of noise-robust automatic speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 745–777, Apr. 2014.

[4] T. Hori, S. Araki, T. Yoshioka, M. Fujimoto, S. Watanabe, T. Oba, A. Ogawa, K. Otsuka, D. Mikami, K. Kinoshita, T. Nakatani, A. Nakamura, and J. Yamato, "Low-latency real-time meeting recognition and understanding using distant microphones and omni-directional camera," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 2, pp. 499–513, Feb. 2012.

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

[6] E. Vincent, S. Watanabe, A.A. Nugraha, J. Barker, and R. Marxer, "An analysis of environment, microphone and data simulation mismatches in robust speech recognition," *Computer Speech and Language*, vol. 46, pp. 535–557, Nov. 2016.

[7] H. Erdogan, T. Hayashi, J.R. Hershey, T. Hori, C. Hori, W.-N. Hsu, S. Kim, J.L. Roux, Z. Meng, and S. Watanabe, "Multi-channel speech recognition: LSTMs all the way through," in *Proc. of The 4th Intl. Workshop on Speech Processing in Everyday Environments (CHiME 2016)*, 2016.

[8] T.H. Dat, N.W.Z. Terence, S. Sivadas, L.T. Tuan, and T.A. Dung, "The I2R system for CHiME-4 challenge," in *Proc. of The 4th Intl. Workshop on Speech Processing in Everyday Environments (CHiME 2016)*, 2016.

[9] J. Du, Y.-H. Tu, L. Sun, F. Ma, H.-K. Wang, J. Pan, C. Liu, J.-D. Chen, and C.-H. Lee, "The USTC-iFlytek system for CHiME-4 challenge," in *Proc. of The 4th Intl. Workshop on Speech Processing in Everyday Environments (CHiME 2016)*, 2016.

[10] T. Menne, J. Heymann, A. Alexandridis, K. Irie, A. Zeyer, M. Kitza, P. Golik, I. Kulikov, L. Drude, R. Schlüter, H. Ney, R. Haeb-Umbach, and A. Mouchtaris, "The RWTH/UPB/FORTH system combination for the 4th CHiME challenge evaluation," in *Proc. of The 4th Intl. Workshop on Speech Processing in Everyday Environments (CHiME 2016)*, 2016.

[11] J.D. Williams, "Exploiting the ASR N-Best by tracking multiple dialog state hypotheses," in *Proc. Interspeech*, 2008, pp. 191–194.

[12] S. Young, M. Gašić, B. Thomson, and J.D. Williams, "POMDP-based statistical spoken dialogue systems: A review," *Proc. IEEE*, vol. 101, no. 5, pp. 1160–1179, Nov. 2016.

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

[14] 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.

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

[16] Y. Tachioka and S. Watanabe, "A discriminative method for recurrent neural network language models," in *Proc. ICASSP*, 2015, pp. 5386–5389.

[17] T. Hori, C. Hori, S. Watanabe, and J.R. Hershey, "Minimum word error training of long short-term memory recurrent neural network language models for speech recognition," in *Proc. ICASSP*, 2016, pp. 5990–5994.

[18] B. Roark, M. Saraclar, and M. Collins, "Discriminative n-gram language modeling," *Computer Speech and Language*, vol. 21, no. 2, pp. 373–392, Apr. 2007.

[19] F.J. Och, "Minimum error rate training in statistical machine translation," in *Proc. ACL*, 2003, pp. 160–167.

[20] M. Collins and T. Koo, "Discriminative reranking for natural language parsing," *Computational Linguistics*, vol. 31, no. 1, pp. 25–70, Mar. 2005.

[21] T. Oba, T. Hori, A. Nakamura, and A. Ito, "Round-robin duel discriminative language models," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 4, pp. 1244–1255, May 2012.

[22] D.E. Knuth, *The Art of Computer Programming, Volume 3: Sorting and Searching, Second Edition*, Addison-Wesley, 1998.

[23] I. Sutskever, O. Vinyals, and Q.V. Le, "Sequence to sequence learning with neural networks," in *Proc. NIPS*, 2014, pp. 3104–3112.

[24] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent NN: First results," in *Proc. Deep Learning and Representation Learning Workshop: NIPS 2014*, 2014.

[25] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. ICASSP*, 2016, pp. 4960–4964.

[26] K. Maekawa, "Corpus of spontaneous Japanese: its design and evaluation," in *Proc. Workshop on Spontaneous Speech Processing and Recognition (SSPR)*, 2003, pp. 7–12.

[27] Preferred Networks, "Chainer: A flexible framework for neural networks," https://chainer.org/.

[28] T. Yoshioka, K. Ohnishi, F. Fang, and T. Nakatani, "Noise robust speech recognition using recent developments in neural networks for computer vision," in *Proc. ICASSP*, 2016, pp. 5730–5733.

[29] 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.

[30] A. Ogawa and T. Hori, "Error detection and accuracy estimation in automatic speech recognition using deep bidirectional recurrent neural networks," *Speech Communication*, vol. 89, pp. 70–83, May 2017.

[31] D.P. Kingma and J.L. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.

[32] G. Neubig, M. Morishita, and S. Nakamura, "Neural reranking improves subjective quality of machine translation: NAIST at WAT2015," in *Proc. WAT*, 2015, pp. 35–41.

[33] J. Niehues, E. Cho, and A. Waibel T.-L. Ha, "Analyzing neural MT search and model performance," in *Proc. The 1st Workshop on Neural Machine Translation*, 2017, pp. 11–17.