This paper introduces a method to train an error-corrective model for Automatic Speech Recognition (ASR) without using audio data. In existing techniques, it is assumed that sufficient audio data of the target application is available and negative samples can be prepared by having ASR recognize this audio data. However, this assumption is not always true. We propose generating probable N-best lists, which the ASR may produce, directly from the text data of the target application by taking phoneme similarity into consideration. We call this process “Pseudo-ASR”. We conduct discriminative reranking with the error-corrective model by regarding the text data as positive samples and the N-best lists from the Pseudo-ASR as negative samples. Experiments with Japanese call center data showed that discriminative reranking based on the Pseudo-ASR improved the accuracy of the ASR.

Index Terms—Large Vocabulary Continuous Speech Recognition, Discriminative Reranking, Error-corrective Model, Call Center Application.

1. INTRODUCTION

In order to boost the accuracy of Automatic Speech Recognition (ASR), discriminative training of a language model (LM) and discriminative reranking with an error-corrective model have been widely studied [1, 2, 3, 4, 5, 6]. In these studies, it is assumed that sufficient audio data of the target application is available and the negative samples are prepared by having ASR recognize this audio data. However, this assumption is not always true. Since building an acoustic model (AM) is expensive and time-consuming, once we build an AM for a certain environment, such as a telephony environment, we sometimes reuse it for several applications. In this case, sufficient audio data of the target application is not collected and we can’t prepare negative samples based on it. As regards an LM, we need to collect a text corpus of the target application and build a new LM for each application.

In this paper, we consider situations in which we are introducing a new application by building an application-specific LM and using an existing AM. To train an LM discriminatively or to estimate an error-corrective model, we need to prepare negative samples from the text data that was collected to build the LM, because the audio data of the target application is not available. There are methods that generate negative samples by using only the text data [9, 10]. However, the negative samples generated without considering any acoustic information are not suitable as the negative samples by the ASR. We present a method to generate probable N-best lists, which the ASR may produce, directly from the correct word sequences by taking phoneme similarity into consideration. In other words, this process simulates the ASR and therefore we call this simulation the “Pseudo-ASR”[11].

In the earlier paper, we showed that the Pseudo-ASR helped in discriminative training of the LM [11]. In this paper, we focus on discriminative reranking by leveraging the results of the Pseudo-ASR. We estimate the error-corrective model by regarding the correct word sequences as positive samples and the N-best lists from the Pseudo-ASR as negative samples. We show that the error-corrective model estimated without using the audio data improves the accuracy of the ASR for Japanese call center data. Then we compare this improvement with the results of discriminative training of the LM.

2. DISCRIMINATIVE RERANKING

We briefly revisit discriminative reranking with the error-corrective model. Given the speech signal $\mathbf{X}$, we denote the N-best list from the ASR as $\mathbf{NB}(\mathbf{X})$. In discriminative reranking, we select the word sequence $\mathbf{W}$ that satisfies Equation (1):

$$W = \arg\max_{\mathbf{w} \in \mathbf{NB}(X)} \{ \log P_{ASR}(\mathbf{w}|\mathbf{X}) + \lambda (\alpha \cdot \Phi(\mathbf{w})) \}$$

The first term $P_{ASR}(\mathbf{w}|\mathbf{X})$ is the recognition score. $\lambda$ is the scaling parameter. The second term $\alpha \cdot \Phi(\mathbf{w})$ is the error-corrective model whose score is calculated as the inner product of the weight vector $\alpha$ and feature vector $\Phi(\mathbf{w})$. The function $\Phi$ converts the word sequence $\mathbf{w}$ to the $F$-dimensional feature vector $\{\phi_1(\mathbf{w}), \ldots, \phi_F(\mathbf{w})\}$. Each function $\phi_i$ represents a feature. The $n$-gram counts in the word sequence $\mathbf{w}$ have been widely used as the features. Note that any arbitrary function other than the $n$-gram counts can be used as $\phi_i$.

Based on the defined features, we estimate the weight vector $\alpha$ by regarding the erroneous recognized results for the utterances as the negative samples and the correct transcriptions as the positive samples. For the estimation algorithm for $\alpha$, “Averaged Perceptron” [12] has been widely used. In addition, we use the conventional “Perceptron” [13] and the recently proposed “Confidence-Weighted Linear Classification” [14].

As we pointed out in Section 1, we assume that there is a shortage of audio data for the target application. Therefore we need to prepare the negative samples in some way. We propose using the Pseudo-ASR to obtain the required negative samples. The details of the Pseudo-ASR appear in the next section.

3. PROPOSED METHOD

In order to estimate the error-corrective model when sufficient audio data is not available, we use the Pseudo-ASR that generates N-best lists directly from the word sequences. In this section, we describe the Pseudo-ASR.
3.1. Pseudo-ASR

Fig. 1 shows an intuitive flow of the Pseudo-ASR:

Stage 1. A word sequence “w₁,w₂” is converted into a phone sequence “p₁₁ · · · p₁₄p₂₁ · · · p₅₆” by consulting the lexicon.

Stage 2. Similar phones whose similarities are estimated from the AM are added with their probabilities to the phone sequence. For example, the phones “p₁₃” and “p₁₄” that are similar to “p₁₃” will be added.

Stage 3. Combined with the lexicon that converts a phone sequence into a word and the LM that assigns a probability to each word sequence, a word graph is generated from the phone sequence.

Stage 4. By conducting a search over the word graph, an N-best list of word sequences is produced.

We explain each stage of the Pseudo-ASR from the viewpoint of its implementation. In Stage 1, we convert a correct word sequence into a phone sequence by consulting the lexicon. We express a phone sequence as a Finite State Acceptor (FSA) \( PS \). Stage 2 and Stage 3 are implemented as the composition of several Weighted Finite State Transducers (WFSTs). Equation (2) is calculated for each input phone sequence \( PS \) and a Viterbi search is conducted over \( \mathcal{WG} \), producing the N-best list.

\[
\mathcal{WG} = \left( ( PS \circ PP ) \circ \mathcal{LX} ) \circ \mathcal{LM} \right).
\]

(2)

\( PP \) : Phone to Phone WFST
\( \mathcal{LX} \) : Lexicon
\( \mathcal{WG} \) : Word Graph
\( \mathcal{LM} \) : LM

Stage 2 corresponds to the composition of \( ( PS \circ PP ) \) and Stage 3 to the remaining compositions. Stage 4 is the search over the \( \mathcal{WG} \) generated in Stage 3. We now explain how each WFST is prepared.

\( PP \) : Phone to Phone WFST

This WFST adds phones to the \( PS \) based on the similarities between the phones in the \( PS \) and the other phones. Intuitively, similar phones are more likely to be misclassified in the ASR. In other words, this WFST simulates the AM of the ASR.

First, we estimate the probability \( \text{prob}(p_j|p_i) \) that the phone \( p_i \) is replaced with \( p_j \) based on the similarity between the phones [11]. Based on this probability, we construct a self-loop WFST. A subset of this WFST is shown in Fig. 2. The transition “\( p_i : p_j / \text{prob}(p_j|p_i) \)” means that this WFST accepts \( p_i \) and outputs \( p_j \) with the probability \( \text{prob}(p_j|p_i) \).

We handle the insertion and the deletion of a phone by using the phone of silence \( p_{Sil} \). The transition “\( p_i : \epsilon / \text{prob}(p_{Sil}|p_i) \)” expresses the deletion of \( p_i \), meaning that this WFST accepts \( p_i \) and outputs no phone with the probability \( \text{prob}(p_{Sil}|p_i) \). The transition “\( \epsilon : p_i / \text{prob}(p_i|p_{Sil}) \)” expresses the insertion of \( p_i \), meaning that this WFST accepts no phone and outputs \( p_i \) with the probability \( \text{prob}(p_i|p_{Sil}) \).

To reduce the computational costs, we limit the number of pairs of phones included in the \( PP \). First, we sort all of the pairs in descending order of \( \text{prob}(p_j|p_i) \). Then we select the top \( C \) pairs.

Since we assume that the audio data of the target application is not available, we resorted to the phoneme similarity to prepare the \( PP \). Using a phonetic confusion matrix from the phone recognition statistics can be an alternative if we can prepare the development set [15, 16, 17].

\( \mathcal{LX} \) : Lexicon

This WFST converts a phone sequence into a word. An \( \mathcal{LX} \) is constructed from the lexicon. For example, this WFST accepts the input phone sequence \( \mathcal{PS} \) and outputs the corresponding word “S P I Y C H”.

\( \mathcal{LM} \) : LM

This WFST assigns probabilities to word sequences. An \( n \)-gram back-off LM can be represented as an WFST [18].

\( \mathcal{WG} \) : Word Graph

With the composition (2), a word graph \( \mathcal{WG} \) is constructed. Then by conducting a Viterbi search over this word graph, the N-best list for the input phone sequence \( PS \) is produced.

3.2. Comparison with Standard ASR

In order to compare the Pseudo-ASR and the standard ASR, we briefly explain how the standard ASR produces the N-best list. Given the speech signal \( X \), the standard ASR produces the \( h \)-th best hypothesized word sequence \( W_h \) that satisfies Equation (3):

\[
W_h = \arg \max_{w \neq W_1, \ldots, W_{h-1}} \left( \beta \log P_{AM} (w|X) + \log P_{LM} (w) \right)
\]

(3)

Here \( P_{AM} \) is the AM probability, \( P_{LM} \) is the LM probability, and \( \beta \) is the inverse of the LM weight. In the proposed Pseudo-ASR, we estimate the first term of the right side of Equation (3) by multiplying the probabilities “\( \text{prob}(p_j|p_i) \)” of the output phone sequence.
Table 1. Coverage ratios of features in 100-best from baseline ASR by those in 100-best from Pseudo-ASR.

<table>
<thead>
<tr>
<th>Feature</th>
<th>1-gram</th>
<th>2-gram</th>
<th>3-gram</th>
<th>D2-gram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage</td>
<td>0.977</td>
<td>0.742</td>
<td>0.677</td>
<td>0.838</td>
</tr>
</tbody>
</table>

4. EXPERIMENT

We conducted a preliminary experiment that confirms the behavior of the Pseudo-ASR. Then we conducted ASR experiments to verify whether the error-corrective model based on the Pseudo-ASR improves the accuracy.

4.1. Experimental Setup

We conducted the experiment using Japanese call center data. We randomly selected 8 calls and used the utterances of the agents as the test data. This test data included 3 male and 5 female agents. The number of utterances was 886 and their total duration was 44.9 minutes. The number of the characters in the transcribed text was 15,323.

We used the existing AM of 57 phones with 5k context-dependent states and 200k Gaussians for the telephone environment, estimated the 3-gram LM with modified Kneser-Ney smoothing [19] from the corpus that was collected for the target application, and prepared a lexicon of 20,652 words with 22,132 pronunciations. The baseline ASR is composed of these AM, LM, and lexicon.

4.2. Performance of Pseudo-ASR

The performance of the error-corrective model is affected by how close the N-best lists from the Pseudo-ASR are compared to those from the standard ASR, because the error-corrective model is estimated with the N-best lists from the Pseudo-ASR. In order to examine how the Pseudo-ASR works, we conducted a preliminary experiment. We selected 275 utterances from the test utterances. We had the baseline ASR produce the 100-best lists from the utterances and had the Pseudo-ASR produce the 100-best lists from the transcriptions of the same utterances. Then we investigated how well the 1-grams, 2-grams, 3-grams, and 1-gap distant 2-grams (D2-gram)3 included in the 100-best lists from the baseline ASR are covered by those from the Pseudo-ASR. The coverage ratios are shown in Table 1. Because the Pseudo-ASR and the baseline ASR are not the same, the coverage ratios were smaller than 1. However, considering that the Pseudo-ASR didn’t use any audio data to generate the N-best lists, good coverage was achieved. Therefore the N-best lists from the Pseudo-ASR are promising as negative samples for the error-corrective model.

4.3. Flow of Experiment

Fig. 3 shows an overview of the experiment and the numbers in Fig. 3 correspond to the following descriptions. (1) First we prepared each WPST for the Pseudo-ASR. (2) We randomly selected T sentences (correct word sequences) from the corpus and had the Pseudo-ASR process them. (3) Then we estimated the error-corrective model by regarding the correct word sequences as positive samples and the N-best lists from the Pseudo-ASR as negative samples. (4) Finally we had the baseline ASR process the test utterances and reranked the N-best lists according to the error-corrective model.

For the Pseudo-ASR, we set the number of pairs of the phones in PP to C = 500 and the size of the N-best list to N = 100. We increased T, the number of correct word sequences used to estimate the error-corrective model, from 5,000 (5k) to 25,000 (25k). For the feature vector Φ, we used the counts of 1-grams, 2-grams, 3-grams, and D2-grams.

4.4. Evaluation and Discussion

First we explain the criterion for evaluation. To measure the ASR accuracy, we used the Character Error Ratio (CER). This is because of the ambiguity in word segmentation in Japanese. For example, “東京都知事” (Governor of Tokyo) can be segmented into words in four ways: (1) “東京都知事”, (2) “東京都 / 知事”, (3) “東京 / 都知事”, and (4) “東京 / 都 / 知事”. In all cases, the same characters are used and the number of the characters remains 5. However, the number of the words seems to change from 1 to 3 because of the ambiguity and the Word Error Rate (WER) fluctuates accordingly. Therefore, the CER is a suitable criterion in Japanese.

4.4.1. CER with Various Numbers of Correct Word Sequences and 3 Algorithms

First, we increased the number T of the correct word sequences used to estimate the error-corrective model from 5k to 25k and examined the CER. In this case, we used “Confidence-Weighted Linear Classification”. Fig. 4 shows the CER as a function of the scaling parameter λ for various T. Note that “Baseline” is the CER when discriminative reranking was not used and in this case the CER was 20.03%. As the number T increased toward 20k, the CER decreased. The best CER among all of the conditions was 18.98% when T was 20k. When T reached 25k, the CER deteriorated. Over-fitting to the correct word sequences seems likely as the cause of this degradation.

Second, to confirm that the N-best lists from the Pseudo-ASR are useful to estimate the error-corrective model regardless of the algorithm used, we tested 3 algorithms, “Perceptron”, “Averaged Perceptron”, and “Confidence-Weighted Linear Classification”. In these experiments, we held T at 20k, which had the best accuracy when changing T. Fig. 5 shows the CER as a function of λ for the 3 algorithms. The CER decreased for every algorithm. When using “Confidence-Weighted Linear Classification”, the CER reached 18.98%, which was equivalent to a 5.24% relative error reduction from the baseline.

In these experiments, we didn’t change the size N of the N-best list and the feature vector Φ. There may be room for improvement by changing N or exploring features other than n-gram counts [2]. For the scaling parameter λ, we tried various values. We need to undertake a study to determine it by using the development set.

3For the word sequence \(w_1 w_2 w_3\), the 1-gap distant 2-gram is “\(w_1 w_3\)”.

Fig. 3. Flow of experiment.
4.4.2. Comparison with Discriminative Training of LM

In the earlier paper, we showed that the Pseudo-ASR helped in discriminative training of the LM [11]. By using the same baseline LM and the same lexicon as we used in this paper, discriminative training decreased the CER from 30.2% to 29.4% on the same test data, which was equivalent to a 2.64% relative error reduction. The discriminative reranking reported in this paper achieved a 5.24% relative error reduction. This is because the error-corrective model can use any arbitrary feature, such as the 1-gap distant n-grams that the n-gram LM can’t handle directly.

5. CONCLUSION

When we introduce a new application by collecting a new text corpus and reusing an existing AM, we devised a new framework so that the error-corrective model can be estimated without using any audio data. In this framework, by leveraging the Pseudo-ASR, the correct word sequences are converted to the probable N-best lists which the standard ASR may produce.

In a preliminary experiment, we confirmed that the Pseudo-ASR produced N-best lists similar to those from the standard ASR, though the Pseudo-ASR didn’t require any audio data. From the experiments with a Japanese call center data, we confirmed that even though the audio data of the target application was not available, the error-corrective model estimated with the N-best lists from the Pseudo-ASR improved the accuracy of the ASR. We saw improvements regardless of the algorithms we used to estimate the error-corrective model. In the best case, the CER for the test data decreased from 20.03% with the baseline ASR to 18.98%, which was equivalent to a 5.24% relative error reduction. This error reduction was statistically significant at the 0.5% level.

6. ACKNOWLEDGMENT

We thank Abhinav Sethy and Bhuvana Ramabhadran of IBM T.J. Watson Research Center for their valuable comments.

7. REFERENCES