Confidence Estimation for Speech Recognition Systems using Conditional Random Fields Trained with Partially Annotated Data

Sheng Li[1], *Xugang Lu*[2], *Shinsuke Mori*[1], *Yuya Akita*[1], *Tatsuya Kawahara*[1]

[1]Kyoto University, Sakyo-ku, Kyoto 606-8501, Japan

[2]National Institute of Information and Communications Technology, Kyoto, Japan

Email: lisheng@sap.ist.i.kyoto-u.ac.jp

Abstract

Conditional random fields (CRF) can generate high-quality confidence measure scores (CMS) for speech recognition systems. However, like many other real-world machine learning tasks, there are only limited annotated data for training but always abundant unlabeled data, which requires too much human efforts and expertise to annotate. To address this issue, we use a scheme of CRF training for ASR confidence estimation, which does not require full annotation transcripts but exploits partially annotated data. We use multiple ASR systems and other freely accessible resources (e.g. caption texts) to generate partially annotated data. Compared with only using a small amount of annotated data and totally using automatically generated unfaithful annotations, the CMS can be enhanced by our proposed method.

Index Terms: speech recognition, confidence measure score, conditional random fields

1. Introduction

The confidence measure score (CMS) indicates the reliability of hypothesis words of automatic speech recognition (ASR) systems. High-quality CMS can improve the data selection for unsupervised acoustic model training [1], MLLR speaker adaptation [2], system combination [3] and various spoken language processing (SLP) applications [4].

A number of approaches have been proposed in the area of confidence estimation [5]. In the conventional methods, CMS is estimated as a posterior probability of a word or an utterance given the acoustic signal through ASR lattices [6], N-best lists [7], word-trellis [8] or confusion network [9], or minimum Bayes risk (MBR) decoding [10, 11]. However, these methods are heavily influenced by the hypothesis size. They are enhanced by using a piecewise linear mapping over decision tree (DT) boundaries [12] or using classifiers trained on a set of predictor features [5, 13].

Conditional random fields (CRF) models [14], which can combine multiple sources such as acoustic, lexical, linguistic and semantic features, with contextual information, can effectively enhance CMS [15, 16]. There are many works in recent years focusing on how to enhance the conventional CRF for deriving robust CMS. In [16], CRF is designed to support continuous features. Hidden variables have also been introduced to the CRF model in [17]. Other studies, e.g. [18] estimated the CMS using CRF models on the confusion networks, and showed improvement in the performance.

However, like many other real-world machine learning tasks, there are only limited annotated data for training but always abundant unlabeled data, which requires too much human efforts and expertise to annotate. To address this issue, a scheme of CRF training, which does not require full annotation transcripts but exploits partially annotated data, has been studied for part-of-speech (POS) tagging, word segmentation [19, 20], named entity recognition tasks [21]. In this method, conditional probabilities over partially annotated data are formulated. Training is achieved by the modification of the learning objective function, incorporating partial annotation likelihood, so that a single model can be trained consistently with a mixture of full and partial annotation [22].

In this paper, we focus on the training CRF-based ASR confidence estimator for speech recognition systems with only limited annotated data and abundant partially annotated data generated by multiple ASR systems and other freely accessible resources (e.g. caption texts). The experiments show that our proposed method can effectively enlarge the training set and enhance the quality of CMS.

In the remainder of this paper, a comprehensive study of training CRF model with partial annotation is formulated in Section 2. Next, we will describe how to generate partial annotation in Section 3. Then, our implementation and the experimental results are presented in Section 4. Finally, the paper is concluded in Section 5.

2. Training CRF with partial annotation

2.1. Full and partial annotations

Fig. 1 and Fig. 2 show examples of full and partial annotations, respectively. In these figures, “T” and “F” stand for the “true” and “false” of the recognized characters. The label sequence is demonstrated as a path consisting of nodes and arrows. By choosing one label for each hypothesis character, we can get full label sequence {(true) → (true) → (false) → (true) → (true) → (true) → (false)…} as shown in Fig. 1.

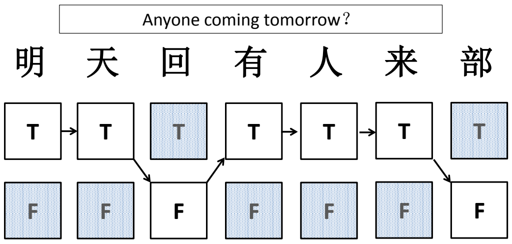

Fig. 1 *Example of fully annotated utterance. (Label sequence is in white color.)*

In the case of partial annotation, instead of assigning each hypothesis character a symbolic label, we assign a non-empty subset of the label space {true, false} to each hypothesis character. In Fig. 2, the label sequence is as follows: {(true) → (true) → (false) → (true, false) → (true, false) → (true) → (true)…}.

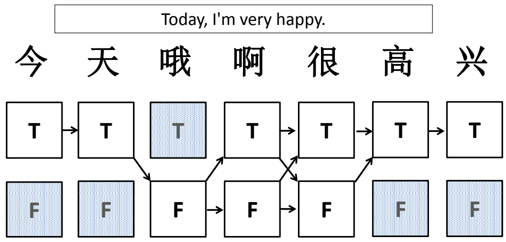

Fig. 2 *Example of partially annotated utterance. (Label sequence is in white color.)*

2.2. Train CRF model with full annotations

A CRF is a discriminative model which estimates the conditional probability. Let $\mathbf{y} = (y_1, y_2, \ldots, y_N)$ be a label sequence given the input feature sequence $\mathbf{x} = (x_1, x_2, \ldots, x_N)$, where N is the sequence length and $y_i \in$ {true, false}. This conditional probability is written as the normalized log-linear function as Equation (1).

$$p_\theta(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z_\theta(\mathbf{x})} \exp\left(\sum_k \lambda_k f_k(\mathbf{x},\mathbf{y})\right) \tag{1}$$

$$Z_\theta(\mathbf{x}) = \sum_{\mathbf{y}} \exp\left(\sum_k \lambda_k f_k(\mathbf{x},\mathbf{y})\right) \tag{2}$$

where $\theta = (\lambda_1, \lambda_2, \ldots, \lambda_k)$ are model parameters, f_k is the k-th feature function, and $Z_\theta(\mathbf{x})$ is the probability normalizer.

For fully-annotated training data, learning of CRF is to maximize the log-likelihood over all the training data as:

$$\theta^* = \arg\max_\theta \mathcal{L}(\theta) \tag{3}$$

$$\mathcal{L}(\theta) = \sum_{p=1}^{N} \log p_\theta\left(\mathbf{y}^{(p)} \mid \mathbf{x}^{(p)}\right) \tag{4}$$

Both the likelihood and its gradient can be calculated by performing the forward-backward algorithm [24] and the sequence optimization algorithms can be used to learn the model parameters, e.g. Limited Memory-BFGS [25].

2.3. Train CRF model with partial annotations

We use the method in [22] and model conditional probabilities over partially annotated data. Training is achieved by modification to the learning objective function, incorporating partial annotation likelihood, so that a single model can be trained consistently with a mixture of full and partial annotation.

As we discussed in Section 2, the possible labels that correspond to the partial annotation as $\boldsymbol{L} = (L_1, L_2, \ldots, L_N)$, where each L_i is a non-empty subset of the label space {true, false} that corresponds to the set of possible labels for feature $\mathbf{x}_i$. Let $\mathbf{Y}_{\boldsymbol{L}}$ be the set of all possible label sequences where $\forall \mathbf{y} \in \mathbf{Y}_{\boldsymbol{L}}, y_i \in L_i$. The conditional probability of $\mathbf{Y}_{\boldsymbol{L}}$ can be modeled as

$$p_\theta(\mathbf{Y}_L \mid \mathbf{x}) = \frac{1}{Z_\theta(\mathbf{x})} \sum_{\mathbf{y} \in \mathrm{Y}_L} \exp\left(\sum_k \lambda_k f_k(\mathbf{x},\mathbf{y})\right) \tag{5}$$

The normalizer $Z_\theta(\mathbf{x})$ is in the same format as in Equation (2). If each element in $\mathbf{Y}_{\boldsymbol{L}}$ is constrained to one single label, the CRF model in Equation (5) will roll back to Equation (1). So we can get a unified framework to train CRF models with both fully and partially annotated data. The log marginal probability of $\mathbf{Y}_{\boldsymbol{L}}$ over N partially annotated training examples can be formalized as follows.

$$\theta^* = \arg\max_\theta \mathcal{L}(\theta) \tag{6}$$

$$\mathcal{L}(\theta) = \sum_{p=1}^{N} \log p_\theta(\mathbf{Y}_L \mid \mathbf{x}) \tag{7}$$

By introducing modification to the forward-backward algorithm [22] with the same optimization algorithms, we can learn the model parameters.

3. Full and partial annotation generation

3.1. Full annotation generation using reference

We generate an ASR hypothesis (1-best) using the ASR system as shown in Fig. 3. By referring to the faithful annotation (reference) after the character-level alignment, we can extract the positive (***T***) and negative (***F***) labels on character level for CRF models training.

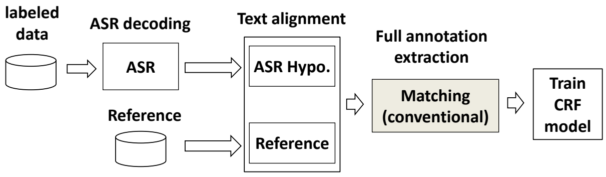

Fig. 3 *Extraction of the full label by matching.*
*(Matching: "**T**" and Mismatching: "**F**".)*

3.2. Partial annotation generation using multiple ASR hypotheses and other accessible texts

Here we propose a voting mechanism to generate a partial annotation for unlabeled data set as shown in Fig. 4. We get the hypotheses from multiple ASR systems. The voting score is calculated from the multiple ASR hypotheses and other accessible texts (e.g. the caption text) by a multi-way character alignment.

The training samples with highest voting scores will be regarded as the positive label (***T***), and the samples with lowest scores are given the negative label (***F***). Others are given a partial label (***T***|***F***).

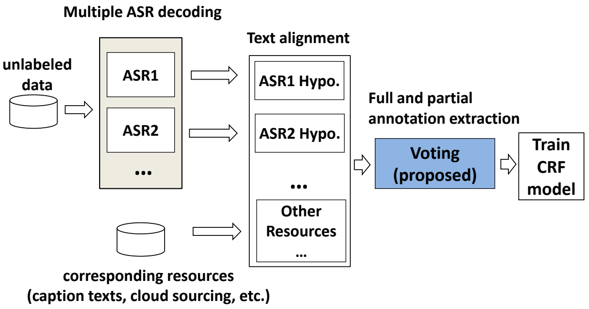

Fig. 4 *Generation of partially annotated data by voting.*

4. Implementation and evaluations

4.1. Data preparation

We made a corpus of Chinese spoken lectures (CCLR) [33]. We selected 58 annotated lectures as the training set (CCLR-SV), 19 annotated lectures as the test set (CCLR-TST) and 12 annotated lectures as the development set (CCLR-DEV). The 126 un-annotated lectures (CCLR-LSV) only with caption text are also used as an additional training set, and we generated the full and partial annotation for it.

4.2. Baseline ASR system and performance

The dictionary for ASR consists of 53K lexical entries from CCLR-SV together with Hub4 and TDT4. The OOV rate on CCLR-TST is 0.368%. The pronunciation entries were derived from the CEDICT[1] open dictionary. We adopt 113 phonemes (consonants and 5-tone vowels) as the basic HMM unit.

A word trigram language model (LM) was built for decoding. We interpolated the 1.07M words text (including faithful transcription texts of CCLR-SV and closed caption texts of CCLR-LSV) with LDC corpora (Hub4 of 0.34M, TDT4 of 4.75M and GALE of 1.03M) and the Phoenix lecture archive[2] (4.12M). The interpolated weights were determined to get the lowest perplexity on CCLR-DEV.

We first build a GMM-HMM system and then a DNN-HMM system. Since the data size of CCLR-SV is not large enough to train a baseline lecture transcription system, we introduced a lightly-supervised training method [23, 34] to enhance the model training by exploiting usable data in another large data set CCLR-LSV with closed caption texts.

The GMM system uses PLP features, consisting of 13 cepstral coefficients (including C0), plus their first and second derivatives, leading to a 39-dimensional feature vector. For each speaker, cepstral mean normalization (CMN) and cepstral variance normalization (CVN) are applied to the features. It is trained with the MPE criterion.

The DNN system uses 40-dimensional filterbank features plus their first and second derivatives with splicing 5 frames on each side of the current frame, and has 1320 nodes as input, 3000 nodes as output and 7 hidden layers with 1024 nodes per layer. The activation function is the sigmoidal function. Training of DNN consists of the unsupervised pre-training step and the supervised fine-tuning step. We use Kaldi toolkit (nnet1) [26]. The SGD uses mini-batches of 256 frames, and a default "Newbob" learning rate schedule. The cross-validation set is held out from the training data by 10%. To accelerate the training time, we use single GPU (Tesla K20m). On this stage, the training is based on the CE criterion, and sequential discriminative training is not conducted. For decoding, we use Julius ver.4.3.1 (DNN version[3]) [27] using the state transition probabilities of the GMM-HMM.

This baseline system achieved an average Character Error Rate (CER) of 24.2% and 27.5% with the MLLR speaker adapted GMM-HMM model, and 22.7% and 25.7% with the DNN-HMM model for CCLR-DEV and CCLR-TST, respectively.

4.3. Full annotation generation using reference

We use the unlabeled data set CCLR-LSV to generate the partial annotation (**LSV-partial**). We can only get unfaithful caption texts instead of references. As an economical choice, we make a three-way character alignment between hypotheses from the baseline DNN (CE) system and the baseline GMM (MPE+MLLR) system and the caption texts. The voting score is the counted. The training samples with the highest scores (voting score=***3***) will be regarded as the positive label (***T***), and the samples with the lowest scores (voting score=***1***) are given the negative label (***F***). Others (voting score=***2***) are given a partial label (***T|F***). The insertion and deletion cases are regarded as a null token.

We also generate a full label of CCLR-LSV (**LSV-full**) for comparison by simply using matching or mismatching information between DNN hypotheses and caption texts. Other full annotation sets listed in Table 1 (**SV**, **TST**, and **DEV**) are generated by matching the ASR hypotheses from the baseline DNN system and the reference texts.

Table 1 *Organization of Annotation for training and testing CRF models.*

Name	Data Sets	#positive ***(T)***	#negative ***(F)***	#partial ***(T|F)***	Label Type
SV	CCLR-SV	161.8K	85.2K	/	Full
LSV-partial	CCLR-LSV	455.3K	119.4K	444.8K	Partial
LSV-full	CCLR-LSV	542.5K	477.0K	/	Full
TST	CCLR-TST	131.1K	43.5K	/	Full
DEV	CCLR-DEV	80.5K	22.5K	/	Full

4.4. Feature design and classifier implementation

A list of features is shown as follows. These features include both acoustic and linguistic information sources. We group these features into two categories: ASR-based features and text-based features. They are listed in Table 2.

Table 2 *Feature design.*

Categorize	Features
ASR-based feature	1. Confidence measure score (**CMS**) [8]. 2. Duration of the current word (**DUR**). 3. Word trigram LM score (**WLM**). 4. Acoustic model score averaged per frame (**AM**). 5. Number of left competing words in the lattice (**NLW**). 6. Number of right competing words in the lattice (**NRW**). 7. Density within word duration (**DEN**).
Text-based feature	1. Lexical entry of current character (**LEX**). 2. Part-Of-Speech for each character unit (**POS**) [28]. 3. 5-gram char LM probability (**CLM**). 4. 5-gram char LM back-off behavior (**BO**).

The ASR-based features are extracted from the word graph during decoding, and simply distributed to each character in the word. They are all numeric features. The text-based features are obtained by syntactic analysis on the character level. **LEX**, **POS**, and **BO** are symbolic features. **CLM** is numeric.

Note that each Chinese character represents a syllable and has a corresponding meaning [32]. We extract the feature from the character level regardless of different word segmentations and OOV problem. With a smaller vocabulary size (about 5K), we can train the CRF model and investigate higher-order language model constraints more efficiently.

Contextual information of two preceding characters and two successive characters are also incorporated in the symbolic features (**LEX**, **POS**, and **BO**).

Integration of numeric features (**CMS**, **DUR**, **WLM**, **AM**, **NLW**, **NRW**, **DEN**, **CLM**) in CRF is not straightforward because CRF implementations process numeric values as symbols. For most of the numeric feature, too many symbols make the feature inefficient. Other than using the method in [29], which modifies the feature function of CRF, our implementation uses the method[4] described in [30] to discretize numeric features; thus grouping together similar values and reducing the number of symbols.

In our experiments, we use the implementation of partial CRF [19], which is based on an open source toolkit CRFSuite[5] and uses the Limited Memory-BFGS algorithm to learn parameters.

The settings for training these CRF models are as follows: Maximize the logarithm of the likelihood of the training data with L1 and L2 regularization terms using the L-BFGS method. The maximum number of iterations for L-BFGS optimization is 100. The cut-off threshold for occurrence frequency of features is 1.

We trained CRF models with various feature sets using CCLR-SV full annotation data and evaluated on CCLR-DEV, as shown in Table 3. The definition of Recall, Precision and F-score are listed as follows:

$$Precision = TP / FP$$

$$Recall = TP / (FP + FN)$$

$$F-score = 2 \times Precision \times Recall / (Precision + Recall)$$

where *TP* is true positives (correct output), *FP* is false positives (false alarm), and *FN* is false negatives (miss).

Table 3 *Classification accuracy on CCLR-DEV.*

	*Positive Label (**T**)*			*Negative Label (**F**)*		
Feature	Recall	Precision	F-score	Recall	Precision	F-score
LEX	0.946	0.834	0.887	0.325	0.628	0.428
POS	0.930	0.822	0.873	0.278	0.525	0.364
CLM	0.961	0.815	0.882	0.219	0.613	0.323
BO	0.895	0.825	0.859	0.322	0.461	0.379
All Text	0.912	0.869	0.890	0.508	0.618	0.557
CMS	0.941	0.833	0.884	0.323	0.605	0.421
DUR	0.968	0.812	0.883	0.197	0.633	0.300
WLM	0.965	0.812	0.882	0.201	0.619	0.304
AM	0.952	0.827	0.885	0.285	0.623	0.391
NLW	0.970	0.810	0.883	0.187	0.634	0.289
NRW	0.964	0.812	0.882	0.202	0.610	0.304
DEN	0.970	0.810	0.883	0.186	0.637	0.288
All ASR	0.921	0.872	0.896	0.517	0.647	0.575
All Features	**0.907**	**0.907**	**0.907**	**0.668**	**0.668**	**0.668**

Among the set of features, the ASR-based features are generally more effective than the text-based features, and the combination of both feature sets shows further improvement. We adopt the complete feature set for following experiments.

4.5. Evaluation

We choose following metrics to evaluate the CMS.

- **Normalized Cross Entropy (NCE):** It assigns the information gain to each of the hypothesis words to assess the quality of the CMS distribution [31]. Higher values of NCE indicate better ASR confidence estimation. Perfect ASR confidence estimates give an NCE of 1. The definition of NCE is as follows:

$$NCE = \left\{ H_{max} + \sum_{correct} log_2(\hat{p}(w)) + \sum_{incorrect} log_2(1 - \hat{p}(w)) \right\} / H_{max}$$

$$H_{\max} = -n \log_2(p_c) - (N - n) \log_2(1 - p_c)$$

  where n is the number of correct hypothesis words, N is the total number of hypothesis words, p_c is the average probability that an output word is correct (=n / N), $\hat{p}(w)$ is the confidence measure of hypothesis word w.

- **Equal Error Rate (EER):** the false alarm rate or the miss rate at the CMS threshold where the false alarm and the miss rate get equal. Lower values of EER indicate better ASR confidence estimation. Perfect ASR confidence estimates give an EER of 0.

We compare the other two CRF models with our proposed method, and they are listed as follows:

- **CRF-baseline:** the model trained by only using full annotation of CCLR-SV (**SV** in subsection 4.3).
- **CRF-full:** the model trained by mixing **SV** and full annotation of CCLR-LSV (**LSV-full** in subsection 4.3).
- **CRF-partial:** the model trained by combining **SV** with partial annotation of CCLR-LSV (**LSV-partial** in subsection 4.3).

We use these three CRF models to generate CMS for the result from the ASR system (baseline DNN-HMM model) on two different evaluation sets as shown in Table 4. The character error rate (CER%) of recognition is 22.7% for CCLR-DEV and 25.7% for CCLR-TST as described in subsection 4.2.

Table 4 *Performance evaluation on CCLR-DEV and CCLR-TST* (by the NIST SCLite scoring tool).

	Annotation Sets for Training		CCLR-DEV		CCLR-TST	
			NCE	EER%	NCE	EER%
CRF-baseline	SV	/	0.360	18.5	0.327	19.0
CRF-full	SV	LSV-full	0.359	18.5	0.324	19.0
CRF-partial	SV	LSV-partial	**0.390**	**18.0**	**0.363**	**18.0**

From Table 4, we observed that the CMS from the **CRF-full** is not improved and even degraded in NCE compared to **CRF-baseline**. This means the full annotation which is automatically generated includes too much noise (uncertainty of labels), and they should not be used for training CRF models directly.

However, our proposed method (**CRF-partial**) can effectively improve the performance on both NCE and EER. The reason is the noisy labels are described as in a probabilistic way by using partial annotation. It can be regarded as a kind of soft weighting in training.

Experiments show that our proposed method can effectively enlarge the training data. We can train the CRF-based classifiers by using partial annotations for data selection and verification in semi-supervised training [35].

In the future, we can efficiently conduct annotating the speech data by selectively annotating a small data set and generating the partial annotations for the rest part. Such active learning technique already has been used in NLP field. Moreover, since data imbalance problem widely exists and always influences the accuracy of classification problem, using partially annotated data could be another possible solution.

5. Conclusions

In this paper, we present a novel CRF model training scheme for ASR confidence estimation with only limited annotated data and abundant partially annotated data. The training data can be effectively enlarged, and experimental evaluations show that the proposed method can enhance the CMS, comparing with only using the small amount of annotated data and entirely using automatically generated unfaithful full annotations.

[1] Available at http://cc-cedict.org/wiki/

[2] Available at http://v.ifeng.com/gongkaike/sjdjiangtang/

[3] Available at http://julius.osdn.jp/en_index.php#latest_version

[4] Available at http://www.irisa.fr/texmex/people/raymond/Tools/tools.html

[5] Available at http://www.chokkan.org/software/crfsuite/

References

[1] K. Yu, M. Gales, L. Wang and P. Woodland, “Unsupervised training and directed manual transcription for LVCSR,” Speech Communication, vol.52, no.7, pp.652–663, 2010.

[2] M. Pitz, F. Wessel, and H. Ney, “Improved MLLR speaker adaptation using confidence measures for conversational speech recognition,” in Proc. Int. Conf. Spoken Language Processing, Beijing, China, Oct. 2000.

[3] J. Fiscus, A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER) ,” In Proc. IEEE- ASRU, 1997.

[4] T. Hazen, S. Seneff, and J. Polifroni, “Recognition confidence scoring and its use in speech understanding systems,” Computer Speech and Language, vol.16, no.1, pp. 49–67, 2002.

[5] H. Jiang, “Confidence measures for speech recognition: A survey,” Speech Communication, vol. 45, no. 4, pp. 455–470, 2005.

[6] T. Kemp and T. Schaaf, “Estimating confidence using word lattices,” in Proc. of EuroSpeech, pp. 827–830, 1997.

[7] F. Wessel, R. Schlüter, K. Macherey, and H. Ney, “Confidence measures for large vocabulary continuous speech recognition,” IEEE Transactions on Speech and Audio Processing, vol. 9, no. 3, pp. 288–298, 2001.

[8] A. Lee, K. Shikano, and T. Kawahara, “Real-time word confidence scoring using local posterior probabilities on tree trellis search,” In Proc. IEEE-ICASSP, Vol.1, pp.793–796, 2004.

[9] L. Mangu, E. Brill, and A. Stolcke, “Finding consensus among words: Lattice-based word error minimization,” In Proc. EuroSpeech, 1999.

[10] H. Xu, D. Povey, L. Mangu, and J. Zhu. “An improved consensus-like method for Minimum Bayes Risk decoding and lattice combination.” In Proc. IEEE-ICASSP, 2010.

[11] V. Goel and W. J. Byrne. “Minimum Bayes-risk automatic speech recognition.” Computer Speech and Language, vol. 14, no. 2, pp. 115–135, 2000.

[12] G. Evermann and P. Woodland, “Posterior probability decoding, confidence estimation and system combination,” In Proc. NIST Speech Transcription Workshop, 2000.

[13] D. Yu, J. Li, and L. Deng, “Calibration of confidence measures in speech recognition,” IEEE Trans. Audio, Speech, and Language Processing , vol. 19, no. 8, pp. 2461–2473, Nov. 2011.

[14] J. Lafferty, A. McCallum, and F. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” In Proc. ICML, 2001.

[15] M. Seigel and P. Woodland, “Combining Information Sources for Confidence Estimation with CRF Models,” In Proc. INTERSPEECH, 2011.

[16] J. Fayolle, F. Moreau, C. Raymond, and G. Gravier, “CRF-based combination of contextual features to improve a posteriori wordlevel confidence measures,” In Proc. INTERSPEECH, 2010.

[17] M. S. Seigel and P. C. Woodland. “Using sub-word-level information for confidence estimation with conditional random field models.” In Proc. INTERSPEECH, 2012.

[18] Z. Ou and H. Luo. “CRF-based confidence measures of recognized candidates for lattice-based audio indexing.” In Proc. IEEE-ICASSP, 2012.

[19] Y. Liu, Y. Zhang, W. Che, T. Liu, and F. Wu, “Domain adaptation for CRF-based Chinese word segmentation using free annotations,” In Proc. EMNLP, pp. 864–874, 2014.

[20] F. Yang and P. Vozila, “Semi-supervised Chinese word segmentation using partial-label learning with conditional random fields,” In Proc. EMNLP, pp. 90–98, 2014.

[21] T. Sasada, S. Mori, T. Kawahara, and Y. Yamakata, “Named entity recognizer trainable from partially annotated data,” In Proc. PACLING, pp.10–17, 2015.

[22] Y. Tsuboi, H. Kashima, S. Mori, H. Oda, and Y. Matsumoto, “Training conditional random fields using incomplete annotations,” in Proceedings of the 22nd International Conference on Computational Linguistics, 2008.

[23] L. Lamel, J.L. Gauvain, and G. Adda, “Lightly Supervised and Unsupervised Acoustic Model Training,” Computer Speech and Language, vol.16, pp.115–129, January 2002.

[24] L. Baum and T. Petrie. 1966, “Statistical inference for probabilistic functions of finite state Markov chains,” The annals of mathematical statistics, pages 1554–1563.

[25] J. Nocedal. “Updating Quasi-Newton Matrices with Limited Storage”. Mathematics of Computation. 35. 151. 773-782. 1980.

[26] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, “The Kaldi speech recognition toolkit,” IEEE-ASRU, 2011.

[27] A. Lee and T. Kawahara, “Recent development of open-source speech recognition engine Julius,” In Proc. APSIPA ASC, pp.131-137, 2009.

[28] M. Shen, H. Liu, D. Kawahara, and S. Kurohashi, “Chinese Morphological Analysis with Character-level POS Tagging,” In Proc. ACL, Short Paper, pp.253-258, Baltimore, USA, 2014.

[29] D. Yu, L. Deng, and A. Acero, “Using continuous features in the maximum entropy model,” Pattern Recognition Letters, vol. 30, no. 14, pp. 1295–1300, 2009.

[30] U. Fayyad and K. Irani, “Multi-interval discretization of continuous attributes for classification learning,” In Proc. IJCAI, pp1022–1027, 1993.

[31] M. Siu, H. Gish, and F. Richardson, “Improved estimation, evaluation and applications of confidence measures for speech recognition,” in Proc. EuroSpeech, 1997.

[32] J. Luo, L. Lamel and J-L. Gauvain, “Modeling Characters versus Words for Mandarin Speech Recognition.” In Proc. IEEE-ICASSP, Taipei, Taiwan, 2009.

[33] S. Li, Y. Akita and T. Kawahara, “Corpus and transcription system of Chinese lecture room,” In Proc. ISCSLP, 2014.

[34] S. Li, Y. Akita, and T. Kawahara, “Discriminative data selection for lightly supervised training of acoustic model using closed caption texts,” In Proc. INTERSPEECH, pp.3526--3530, 2015.

[35] S. Li, Y. Akita and T. Kawahara, “Semi-supervised acoustic model training by discriminative data selection from multiple ASR systems' hypotheses.” IEEE/ACM Trans. Audio, Speech and Language Processing, vol.24, no.9, pp.1520–1530, 2016.