CRF-BASED CONFIDENCE MEASURES OF RECOGNIZED CANDIDATES FOR LATTICE-BASED AUDIO INDEXING[1]

Zhijian Ou, Huaqing Luo

Department of Electronic Engineering, Tsinghua University, Beijing, China
Emails: ozj@tsinghua.edu.cn, luohuaqing@live.cn

ABSTRACT

The use of forward-backward (FB) computation based posterior probabilities as confidence measures (CMs) for all recognized candidates in a lattice seems to be common across various lattice-based audio indexing systems. However, a major limitation with this approach is that its performance for CMs cannot be improved easily, since it relies almost entirely on a single information source - the acoustic and language-model probabilities. In this paper, we propose to formulate computing CMs in the lattice case as a multi-class sequential labeling problem, using conditional random fields (CRFs) as the underlying model. In this approach, various relevant features including the FB posterior probabilities could be combined together. Note that CRFs are well suited to label sequence data and some features are defined over a word sequence. This paper presents how we resolve these two issues in the lattice case, beyond others' previous work in CRF-based CMs for the 1-best case. Once properly implemented, the proposed approach achieves significant performance improvements for both CMs in the lattice case and lattice-based audio indexing.

Index Terms— Confidence measure, CRF, audio indexing

1. INTRODUCTION

There is a tremendous increase of audio data, such as web videos, meeting recordings, voicemails, etc. Audio indexing refers to the task of first processing the audio by automatic speech recognition (ASR), and then indexing the recognized results so as to answer information-retrieval queries from users [1-3]. A typical query is to find in the audio the desired keyword entered by the user.

Note that real-world audio is still a challenge for today's speech-to-text ASR system. So instead of using only the 1-best transcript, modern audio indexing systems search alternative recognized candidates as well [2, 3]. The recognized candidates, either words or subwords, are represented in a graph structure, called *lattice*. For every recognized candidate in the lattice, a posterior probability is calculated from the acoustic and language-model probabilities using the well-known forward-backward (FB) technique [4]. The lattice, or some other further reduced structure, is searched to find the word/subword sequence that matches the query keyword. The posterior probabilities are then used as confidence measures (CMs) of the recognized candidates to sort the matches.

Different audio indexing systems may differ in many aspects such as the indexing unit (word, subword, or hybrid) [3], the specific form of the reduced structures, e.g. time-anchored lattice expansion (TALE) [2], confusion network (CN) [5]. In contrast, the use of FB posterior probabilities as the confidence scores seems to be common across various audio indexing systems. While there are a lot of studies on CMs for speech recognition [6], most of them seek CMs mainly for the recognized 1-best (*the 1-best case*) rather than for all recognized candidates in a lattice (*the lattice case*). Computing CMs for the 1-best case benefits many ASR applications, e.g. to detect recognition errors, assist spoken dialog, but it is insufficient for lattice-based applications, e.g. audio indexing, where computing CMs for the lattice case is required.

How to effectively compute CMs for the lattice case is the main issue studied in this paper. The approach of using FB posterior probabilities as CMs appears to be a good choice. After a forward-backward computation, we obtain all of them. However, a major limitation with this approach is that its performance cannot be improved easily, since it relies almost entirely on a single information source - the acoustic and language-model probabilities. From this perspective, the approach of computing CMs by combining various relevant features is more attractive. In this approach, computing CMs for the recognized 1-best is formulated as a binary classification problem with the posterior probability of labeling 'correct' as the CM for each word. Various features and classification models are proposed, as surveyed in [6]. In a recent work [7], improved performances of CMs for the 1-best case are obtained by introducing (linear-chain) conditional random fields (CRFs) [8] to do sequential labeling and using contextual features.

In this paper, we propose to formulate computing CMs in the lattice case as a multi-class sequential labeling problem, using CRFs as the underlying model. In this approach, various relevant features including the FB posterior probabilities could be combined together. It should be noted that it is not trivial to extend CRF-based CMs from the 1-best case (as shown in [7]) to the lattice case. Two issues need to be resolved. First, (linear-chain) CRFs are probabilistic models suited to label sequence data[2], while the lattice from the ASR decoder is not sequential. Thus we propose to first reduce the lattice to a linearized structure - sausage. This is compatible with audio indexing and could be achieved by either the TALE or the CN method, both of which align word candidates to word positions. Second, some features, e.g. contextual features, are defined over a word sequence. We need to figure out methods to extract such features from the lattice, which is not as straightforward as from the 1-best transcription. This

[1] This work is supported by National Natural Science Foundation of China (61075020) and China 863 (2006AA01Z149).

[2] Most applications of CRFs use linear-chains. Although theoretically, we can define CRFs with general graphical structures, this paper only considers how to use linear-chain CRFs for CMs in the lattice case. We leave the possibility of applying general-structure CRFs for future studies.

paper presents how we resolve the above two issues. Experimental evaluations show that the new approach achieves significant performance improvements for both CMs in the lattice case and lattice-based audio indexing.

This paper is organized as follows. We detail the proposed CRF-based approach for CMs in the lattice case, including the CRF model formulation and the features used in the CRFs in Section 2 and Section 3 respectively. Evaluation results for both CMs and audio indexing are provided in Section 4. Finally, the conclusions are made in Section 5.

2. CRFS FOR CMS IN THE LATTICE CASE

A conditional random field is a conditional distribution $p(q|y)$ with an associated graphical structure, which is often a linear-chain in most cases. The variable y represents the observations/inputs, and the variable q represents the attributes/labels that we wish to predict from the observations. Various *features* are computed based on the input observations. The CRF model affords the use of rich, non-local features of the input observations.

Take the computation of CMs in the 1-best case as an example. The observation sequence y is defined to be the 1-best transcript. Useful features include FB posteriors, part-of-speech, word duration, etc. Contextual features based on these base features can also be supported. The label variables are structured as a linear-chain. Corresponding to the n-th position of the 1-best transcript, there is a binary label variable q_n, taking the values of being 'correct' or 'false'. The posterior $p(q_n = \text{'correct'} \mid y)$ is used as the CM for the n-th word.

Now consider the computation of CMs in the lattice case. Suppose that we take the lattice directly as the observations y, and for each word candidate, we define a binary label variable as in the 1-best case. Then these label variables should interact with each other. The underlying graphical structure is not easily to be determined and certainly become more complicated. We leave the possibility of applying general-structure CRFs for future studies.

In this paper, we propose to first reduce a lattice to a linearized structure - sausage. Then, we could define a CRF over the sausage.

2.1. Reduce lattice to sausage

Reducing a lattice to a sausage is compatible with audio indexing. For audio indexing, the lattice from the ASR decoder is also required to be reduced to some forms of smaller-sized and linearized structure, which is beneficial for indexing and searching. The reduction could be achieved by either the TALE [2] or the CN [5] method, both of which align word candidates to word positions. The TALE method is used in our experiments due to its simplicity.

Fig. 1 shows an example of a lattice and the sausage that is converted from the lattice. A lattice is a weighted directed graph where the arcs represent word candidates with FB posterior probabilities. For the sausage, there is a linear sequence of slices, each of which includes a number of word candidates (also called word arcs).

2.2. Define CRF over sausage

After we obtain the sausage from the lattice, we could define a linear-chain CRF over the linear-structured sausage. Suppose that the sausage has a total number of N slices and there are a total number of K_n word arcs contained in the n-th slice, which are

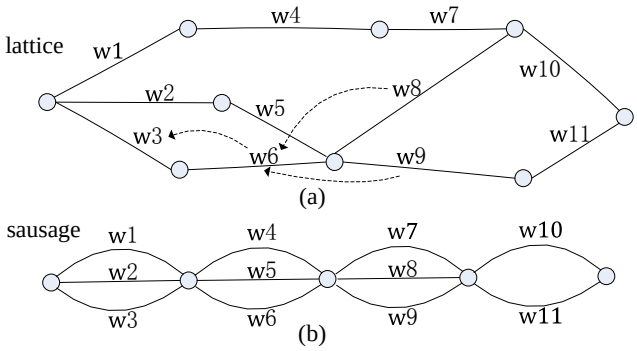

Fig.1: Example of a lattice and the sausage reduced from the lattice

denoted by $ARC_1^n, \cdots, ARC_{K_n}^n$ respectively.

The observation sequence $y_1 \cdots y_N$ is now defined to be the sausage, where y_n represents the n-th slice. Corresponding to the n-th slice, we define a multi-valued label variable q_n that can take on a total number of $K_n + 1$ values, which are $ARC_1^n, \cdots, ARC_{K_n}^n$ plus a special value called 'null'. Labeling q_n with ARC_k^n, $k = 1, \cdots, K_n$, means that the word arc ARC_k^n is correct. Labeling q_n with 'null' means that none of the word arcs contained in the n-th slice is correct. In this manner, the word arcs contained in the same slice compete with each other to be the correct one.

The conditional distribution $p(q|y)$ is defined as follows:

$$p(q|y) \propto exp\left\{\sum_{n=1}^{N} \phi_n(q_n, y) + \sum_{n=2}^{N} \psi_n(q_{n-1}, q_n, y)\right\} \quad (1)$$

Here $\phi_n(q_n, y)$ is the node potential function at position n, which is further computed as

$$\phi_n(q_n, y) = \lambda_{\text{other}} 1(q_n = \text{'null'}) + \bar{\lambda}_{\text{other}} 1(q_n \neq \text{'null'}) + \sum_{k=1}^{K_n} \sum_{f} \left[\lambda^f_{f(ARC_k^n)} 1(q_n = ARC_k^n) + \bar{\lambda}^f_{f(ARC_k^n)} 1(q_n \neq ARC_k^n)\right] \quad (2)$$

Here $f(\cdot)$ denotes a particular (discrete-valued) feature. $f(ARC_k^n)$ represents the specific feature value that is computed for word arc ARC_k^n, e.g. the discretized FB posterior probability of ARC_k^n. $1(\cdot)$ denotes the indicator function, e.g.

$$1(q_n = ARC_k^n) = \begin{cases} 1 & if\ q_n = ARC_k^n \\ 0 & otherwise \end{cases}$$

The summation over f means that we have a number of different features computed for each word arc, which will be described in the next section.

Equ. (2) defines the potential function to be a weighted combination of a set of indicator functions, which is a common practice for applying CRFs. The $\lambda's$ and $\bar{\lambda}'s$ are the weights indexed by the values of various features. The superscript indicates which feature the weight is related to. Consider a particular feature $f(\cdot)$ and suppose that q_n is labeled with ARC_k^n. Then, the corresponding weight $\lambda^f_{f(ARC_k^n)}$ will contribute to the conditional distribution $p(q|y)$. Otherwise (i.e. q_n is labeled with word arcs other than ARC_k^n), the corresponding contrary weight $\bar{\lambda}^f_{f(ARC_k^n)}$ will contribute to the conditional distribution. Depending on whether q_n is labeled with 'null' or not, two feature-independent weights - either λ_{other} or $\bar{\lambda}_{\text{other}}$, are activated, since no features are defined for the pseudo word arc 'null'.

The $\psi_n(q_{n-1}, q_n, y)$ in Equ. (1) is the edge potential function at position n, which is computed as

$$\psi_n(q_{n-1}, q_n, y) = \lambda^e_{e(q_{n-1}, q_n, y)} \quad (3)$$

Here we use only one transition feature $e(q_{n-1}, q_n, y)$ whose value is 1 if the word arcs given by q_{n-1} and q_n (by referring to the sausage y) are successively appeared in a path during ASR decoding and 0 otherwise.

Once all the weights are estimated from training data, we can compute the posterior probability $p(q_n = ARC_k^n \mid y)$ as the CM for word arc ARC_k^n based on the trained CRF model.

3. FEATURES FOR CMS IN THE LATTICE CASE

To achieve good performances of CMs using the CRF-based approach, we need a set of relevant features that are informative to distinguish correctly recognized words from possible recognition errors. A lot of features are studied in the literature for CMs in the 1-best case, from which we select two types of base features – *fbconf* and *lmbb*, which are reported to have the most impact to improve the CM performance [7].

A difference between the 1-best case and the lattice case for computing features is that in the lattice case, the features are needed to be computed for all the word candidates in the sausage, rather than only for the 1-best transcription as in the 1-best case. This will bring some difficulty in generating certain features which are defined over a word sequence, e.g. the *lmbb* feature and all contextual features. In the following, we will present our solution to extract such features from the lattice.

3.1. Base features

First, there is no difficulty in generating *fbconf* feature. The FB posterior probabilities are computed for each word arc in the lattice. In reducing the lattice to the sausage, the FB posterior probabilities are naturally retained for the word arcs in the sausage. The *fbconf* feature is then obtained by discretizing the continuous posterior probability into 6 classes based on K-mean clustering in our study.

In a word sequence, the language model back-off behavior (LMBB) for each word is the largest degree *n* of the current *n*-gram appeared in the chosen language model [7]. The 3-gram language model from our ASR system is used here. This base feature includes 3 main classes ('I1', 'I2', 'I3') and 3 other specific classes ('11', '21', '22') that represent the different cases in the beginning of the word sequence.

The *lmbb* feature is defined over a word sequence. However, during decoding, we can keep track of the best partial path up to every word candidate. The immediate predecessor words are stored for every word candidate in the lattice during lattice generation. For example, in Fig. 1(a), the immediate predecessor word for word candidate $w9$ is $w6$, which has the immediate predecessor word $w3$. So we can determine the *lmbb* feature for word candidate $w9$ by checking the trigram $w3$-$w6$-$w9$ against the trigram language model.

3.2. Contextual features

Using context to enrich base features has shown promising results for CMs [7]. Thus we generate contextual features for both the *fbconf* feature and the *lmbb* feature. The contextual feature is well defined over a word sequence, as detailed in [7]. Specifically, a contextual feature with the 3-gram pattern '-1/0/1' means that it is created by joining the base feature at the previous position, the current position, and the next position in the sequence.

To generate such contextual features, we need to know not only the immediate predecessor word which can be uniquely determined (as explained above for *lmbb* feature generation), but also the immediate successor word which is not necessarily uniquely determined. Our solution is that if there are a number of paths passing through the current word, e.g. $w6$ in Fig. 1(a), we choose the best partial path (as measured by the average likelihood of the partial path normalized by the number of frames) to determine the immediate successor, e.g. $w8$ (if the average likelihood of the partial path '$-w6-w8$' is larger than that of the partial path '$-w6-w9$').

4. EXPERIMENTAL SETUP AND RESULTS

Evaluation experiments are carried out with a large vocabulary Chinese continuous speech recognition system. The benchmarking of this system for 1-best transcription is described in [9]. Here we use the same acoustic features and HMMs as in [9], but update the language model to be a trigram model trained from about 200 million Chinese words mostly coming from Chinese newspapers. A total of about 12 hours of Chinese male spontaneous speech data are used, which is divided into a development and a test set of 6h each (similar to [7]), for which Chinese Character error rates (CCER) of 30.38% and 31.87% are achieved respectively. We use the development set to train the CRF classifiers and the test set to evaluate performance. The CRF++ tool [10] in its original form cannot support the CRF as defined in Section 2, and is modified to do training and inference for the new CRF model.

In the experiments, both the development and test speech data are first decoded to generate Chinese word lattices, which are then converted to Chinese character lattices. The TALE method is used to reduce the character lattices to character sausages. Then, the CMs for all the word candidates in the sausage are computed, either from the classic FB posteriori probabilities (the baseline) or from the posterior probability provided by the CRF classifiers. In the following, we first evaluate the pure CM performance of the proposed CRF-based approach, and then examine its effectiveness in a practical audio indexing system.

4.1. Experiments of confidence measures in the lattice case

We first implement a tool to align the character sausage with the reference character sequence using dynamic programming. Once the optimal alignment is found, we could determine whether an arc in the sausage is correct or not. In this way, the true labels for all the arcs in the sausage are created. For the development data, the labels are used to train the CRF model. For the test data, the labels are used to evaluate the CM performance. Once the confidence has been computed, every arc of the sausage can be tagged as either correct or false, depending on whether its confidence exceeds a certain threshold or not. For all the arcs in the sausage, we compare the tagging with the true label and can find two types of errors, namely false alarm errors and false rejection errors. Finally, we can compute the equal error rate (EER) and plot the detection-error-tradeoff (DET) curve, which are the two standard evaluation metrics for CMs [4].

Note that we have 3 base features - *fbconf*, *lmbb*, and their combination – *fbconf*+*lmbb*. For each base feature, we could use the base feature itself, or use the contextual feature with the 3-gram pattern '-1/0/1'. So there are a total of 6 different configurations of features. For each configuration, we separately train a CRF model and the resulting model is evaluated in the test set. The EERs of the CRF-based classifiers using the 6 different configurations of features are summarized in Table 1.

Table 1: performance comparisons between the baseline and the CRF-based approach, using different configurations of features for CMs (in terms of EER) and audio indexing (in terms of keyword search EER). #weight denotes the total number of weights used in the CRF model.

features used for the CRF		CM EER	keyword search EER	#weight
baseline		42.63%	25.70%	
base	*fbconf*	41.58%	24.83%	9
	lmbb	42.96%	26.50%	9
	fbconf+lmbb	41.20%	24.53%	15
contextual	*fbconf*	40.51%	23.98%	297
	lmbb	41.52%	25.50%	65
	fbconf+lmbb	40.13%	23.24%	359

The main results are similar to [7]. First, the EER obtained using the *fbconf* feature is slightly better than the baseline which uses the continuous FB posterior probabilities. This confirms that the CRF model is powerful for discriminative learning and classification and that the discretization does not affect the quality of the CM. Second, the *lmbb* feature is informative enough to perform only slightly worse than the baseline. Third, using the context feature always results in increased performances. Fourth, significant improvements in EER are obtained by combining *fbconf* and *lmbb*, both in the base case and in the contextual case, which can also be seen from the DET curves in Fig. 2. In conclusion, the (linear-chain) CRF model is successfully applied to compute CMs for all the word candidates in the sausage, performing much better than the baseline.

4.2. Experiments of lattice-based audio indexing

In audio indexing, the sausage is searched to find the character sequence that match the query keyword entered by the user, e.g. '清华'. We need the CMs for all the word candidates in the sausage to sort the matches. The experiments in Section 4.1 evaluate the pure CM performance of the proposed CRF-based approach, which is independent of the keyword set. The following experiments examine its effectiveness in a practical audio indexing system.

The experimental keyword set contains a number of 500 Chinese 2-character noun phrases with a total of 5182 occurrences in the 6h test data. For each keyword, the sausage is scanned to find matches. Given a certain threshold, the found matches for all the keywords are filtered, depending on whether the confidences exceed the threshold or not. Comparing the filtered matches with the true labels, we can compute the EER and plot DET curve. It is shown in Table 1 the keyword search EERs for audio indexing based on the same CRF-based classifiers as used in Section 4.1. It can be seen from Table 1 and Fig. 3 that the advantage of the proposed CRF-based approach for CMs clearly helps to improve the searching accuracies of the audio indexing system.

5. CONCLUSION

Computing CMs in the lattice case means computing CMs for all recognized candidates in a lattice, which is useful for lattice-based applications, e.g. audio indexing. A major limitation with FB computation based posterior probabilities as CMs in the lattice case is that its performance for CMs cannot be improved easily, since it relies almost entirely on a single information source - the acoustic

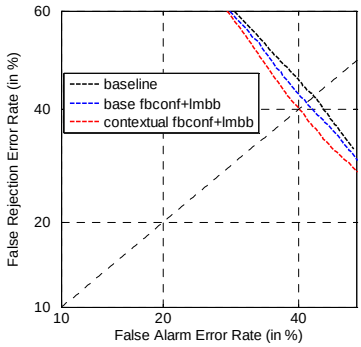

Fig 2: The DET curves for confidence measures

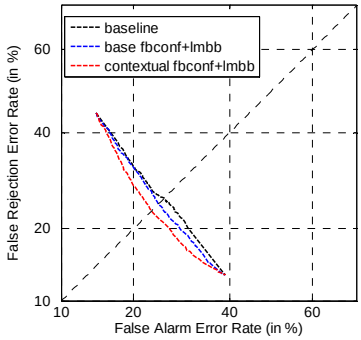

Fig 3: The DET curves for keyword search in audio indexing

and language-model probabilities. In this paper, we propose CRF-based CMs for the lattice case. In this approach, various relevant features including the FB posterior probabilities could be combined together. Once properly implemented, the proposed approach achieves significant performance improvements for both CMs in the lattice case and lattice-based audio indexing.

6. REFERENCES

[1] M. Christel, *et al.*, "Speech and language technologies for audio indexing and retrieval", *Proc. the IEEE*, 2000.

[2] P. Yu, Y. Shi, F. Seide, "Approximate word-lattice indexing with text indexers: time-anchored lattice expansion", *Proc. ICASSP*, 2008.

[3] Y. Pan, L. Lee, "Performance analysis for lattice-based speech indexing approaches using word and subword units", *IEEE Trans. ASLP*, 2010.

[4] F. Wessel, *et al.*, "Confidence measures for large vocabulary continuous speech recognition", *IEEE Trans. SAP*, 2001.

[5] L. Mangu, E. Brill, A. Stolcke, "Finding consensus in speech recognition: word error minimization and other applications of confusion networks", *Computer Speech and Language*, 2000.

[6] H. Jiang, "Confidence measures for speech recognition: a survey", *Speech Communication*, 2005.

[7] J. Fayolle, *et al.*, "CRF-based combination of contextual features to improve a posteriori word-level confidence measures", *Proc. Interspeech*, 2010.

[8] J. Lafferty, A. McCallum, F. Pereira, "Conditional random fields: probabilistic models for segmenting and labeling sequence data", *Proc. ICML*, 2001.

[9] Z. Ou, J. Xiao, "A study of large vocabulary speech recognition decoding using finite-state graphs". *Proc. ISCSLP*, 2010.

[10] http://crfpp.sourceforge.net/