AN INITIAL ATTEMPT TO IMPROVE SPOKEN TERM DETECTION BY LEARNING OPTIMAL WEIGHTS FOR DIFFERENT INDEXING FEATURES

Yu-Hui Chen [#], Chia-Chen Chou [#], Hung-Yi Lee [*], Lin-shan Lee [*]

Department of Electrical Engineering, National Taiwan University [#]
Graduate Institute of Communication Engineering, National Taiwan University [*]

ABSTRACT

Because different indexing features actually have different discriminative capabilities for spoken term detection and different levels of reliability in recognition, it is reasonable to weight the indexing features in the transcribed lattices differently during spoken term detection. In this paper, we present an initial attempt of using two weighting schemes, one context independent (fixed weight for each feature) and one context dependent(different weights for the same feature in different context). These weights can be learned by optimizing a desired spoken term detection performance measure over a training document set and a training query set. Encouraging initial results based on unigrams of Chinese characters and syllables for the corpus of Mandarin broadcast news were obtained from the preliminary experiments.

Index Terms— Spoken Term Detection, SVM-map

1. INTRODUCTION

The need for improved spoken term detection technologies has been growing very rapidly, primarily due to the fast increase of the multimedia content over the Internet, whose central information is very often in the associated audio data. Efficient and convenient use of all those multimedia content relies on the availability of successful retrieval technologies, since people cannot go through all of them one by one. A straightforward approach to this problem is to use speech recognition techniques to translate the spoken segments into text files and then apply text document retrieval techniques on them. However, the relatively poor recognition accuracies for spontaneous speech including out-of-vocabulary (OOV) words under adverse environments usually result in very poor retrieval performance. Combination of word and different subword level information has been proved as an efficient method to improve the performance of spoken term detection, and somehow solve the OOV problem [1, 2]. Also, People have been using multiple hypotheses of speech recognition output, very often in some efficient representation of the lattices generated in the recognition process [3, 4]. Position Specific Posterior Lattices (PSPL) [5], Confusion Networks (CN) [6], and Time-based Merging for Index (TMI) [7] are good examples of such efficient representations. They are referred to as lattice-based approaches in this paper.

Although lattice-based approaches have been shown to provide much more useful information than the one-best recognition output and therefore produce better recall rates, the much more noisy information included in the lattices also inevitably degrades the precision and the ranking of relevant documents. As a result, efficient approaches to de-emphasize those indexing features more likely to be noisy, but emphasize those important indexing features naturally becomes crucial. Although many very successful similar weighting schemes have been developed for text-based information retrieval, they cannot be directly applied to lattice-based voice-based information retrieval due to the quite different nature of the problems. SVM-MAP [8] has been successfully used to estimate feature weights optimizing retrieval evaluation measure, MAP, in spoken document retrieval [9], but only the weights for different types of features (words or different subword units) were estimated.

In this paper, we adopt the concept of learning to rank to develop efficient schemes of weighting different indexing features from the concept of vector space model (VSM) to optimize the performance of lattice-based spoken term detection. VSM has been widely used in both text- and voice-based information retrieval [10, 11, 12, 13, 14]. In this approach, each document and each query are both represented as vectors, with each component of the vectors corresponding to a certain indexing feature (or term), usually a word or a subword unit allowed by the lexicon. The retrieval system then ranks the documents by the dot product between the query vector and the document vector. Because different indexing features have different importance for the spoken term detection task, properly assigning different weights to each individual indexing feature or term is crucial. This can be achieved by defining two functions $W(u,q)$ and $W(u,d)$ as the weights representing the term u in the vectors for the query q and the document d,

$$\vec{q} = \begin{bmatrix} W(u_1,q) \\ W(u_2,q) \\ \vdots \\ W(u_K,q) \end{bmatrix}, \vec{d} = \begin{bmatrix} W(u_1,d) \\ W(u_2,d) \\ \vdots \\ W(u_K,d) \end{bmatrix}, \quad (1)$$

where K is the total number of terms considered. In text retrieval, people usually simply have $W(u,.) = tf(u,.) \times idf(u)$ in (1). $tf(u,.)$ refers to term frequency, and $idf(u)$ refers to inverse document frequency.

In lattice-based spoken term detection, the audio data are divided into segments, referred to as "spoken segments" in this paper. Spoken segments are transcribed into lattices, and then transformed into sausage-like structures such as Position Specific Posterior Lattices (PSPL), which are sequences of clusters and each cluster include a number of arcs of hypothesis of words or subword units. The definition of $tf(u,.)$ and $idf(u)$ in text retrieval can be modified accordingly to consider such lattice-based scenario. The posterior probabilities of a specific term u in a lattice-based structure is usually considered [10, 11, 12, 14] as term frequency, but the definition of inverse document frequency (idf) is not so intuitive in lattice-based structure. idf derived on manual transcription can not work properly on spoken term detection because it does not consider recognition error at all. idf cannot be successfully derived from automatically transcribed lattices either, since incorrectly recognized hypotheses may appear repeatedly in a lattice. In this paper, we try to estimate

the weights of each term to take the role of idf. In this approach, the discriminating capabilities for retrieval and the reliability in recognition for each indexing feature (or term) are jointly considered during learning.

In Sections 2 and 3, we represent the two weighting schemes proposed in this paper, one context independent and one context dependent. We present the experimental results in Section 4. We make concluding remarks in Section 5.

2. PROPOSED APPROACH 1: CONTEXT INDEPENDENT WEIGHTING

We use the posterior probabilities to evaluate term frequency $tf(u,d)$, but we wish to find a weight $M(u)$ for each term, independent of its context, to take the place of $idf(u)$, so $W(u,d)$ in (1) is changed into $tf(u,d) \times M(u)$. We try to find $M(u)$ for each term u in such a way that the performance measure for retrieval is optimized on a training query set and a training document set. In this way, the weight $M(u)$ represent both the discriminative capabilities and the recognition reliability of the term u. We use $\mathcal{M}$ to denote a weight set of all terms, $\mathcal{M} = \{M(u),\ \text{all } u\}$.

The retrieval task has a training stage and testing stage. In the training stage, the weight set is estimated on the training query set and the training document set, which is then used in the testing stage.

2.1. Training Stage

Here, we need to estimate the weight $M(u)$ for each term u. The training data set can be represented as $S = \{q_i, y_i\}_{i=1}^m$. Here the training query set $\mathcal{Q}_{train} = \{q_1, q_2, ..., q_m\}$ includes m training queries, and y_i is the true answer to the query q_i for the training document set $\mathcal{D}_{train}$ with N documents, where $y_i = [y_{i1}, y_{i2}...y_{iN}]$, $y_{ij} \in \{T, F\}$ indicating whether the document d_j in $\mathcal{D}_{train}$ is relevant to the query q_i or not. Let $r_i(\mathcal{M})$ be the ranking list obtained based on the weight set $\mathcal{M}$ when the query q_i is entered. We define a performance evaluation function $E(y_i, r_i)$ which gives the retrieval performance by comparing the obtained rank list r_i with true answer y_i. Below we take Mean Average Precision (MAP) as this function $E(y_i, r_i)$, although some other measures can also be used [8]. So our goal is to find $\mathcal{M} = \{M(u),\ \text{all } u\}$ such that

$$\hat{\mathcal{M}} = \arg\max_{\mathcal{M}} \sum_{i=1}^{m} E(y_i, r_i(\mathcal{M})). \tag{2}$$

It may not be easy to define a training query set. In our task on Mandarin Chinese, both characters and syllables are very useful indexing features. We selected a finite number of most important characters by tf/idf scores evaluated from a text corpus. We try to assign weights to these characters and all syllables (roughly 400 in total). So we enumerated all possible character bi-grams for these character terms and syllable bi-grams for all syllable terms to be used as the training query set.

The Support Vector Machine (SVM) has been found to be helpful in ranking the documents according to their relevance to the query. In particular, SVM-MAP has been shown to be a powerful solution that can separate the best ranking sequence from other possible sequences in terms of MAP with a maximized margin [15]. Here in this research, we solve (2) using SVM-MAP as explained below very briefly. We first defined a feature vector $\Phi(q,d)$ for each document d and each query q, SVM-MAP is then able to find a weight vector $\vec{w}$ which optimizes the upper bound of MAP on a training data set $S = \{q_i, y_i\}_{i=1}^m$ when the dot product $\vec{w} \cdot \Phi(q,d)$ is used to rank the documents [8]. Note that according to VSM we rank the documents based on $\vec{q} \cdot \vec{d} = M(u_1)tf(u_1,d)tf(u_1,q) + M(u_2)tf(u_2,d)tf(u_2,q) + \cdots + M(u_K)tf(u_K,d)tf(u_K,q)$. Therefore when we define

$$\Phi(q_i, d_j) = \begin{bmatrix} tf(u_1,d)tf(u_1,q) \\ tf(u_2,d)tf(u_2,q) \\ \vdots \\ tf(u_K,d)tf(u_K,q) \end{bmatrix}. \tag{3}$$

Clearly the weight vector $\vec{w}$ returned by SVM-MAP is exactly the weight set we need,

$$\mathcal{M} = \vec{w} = \begin{bmatrix} M(u_1) \\ M(u_2) \\ \vdots \\ M(u_K) \end{bmatrix}. \tag{4}$$

2.2. Testing Stage

With the weight set $\mathcal{M}$ estimated above, the retrieval process is natural with a testing spoken document set $\mathcal{D}_{test}$ and a testing query set $\mathcal{Q}_{test}$.

3. PROPOSED APPROACH 2: CONTEXT DEPENDENT WEIGHTING

The above weighting scheme assumes all terms have a fixed weight, but in fact the importance of a term in retrieval may depend on the context. For example, consider two queries, "information retrieval" and "information theory". The term "information" is actually less important in the context of "information retrieval", because many relevant documents may include the term "retrieval", but not necessarily the term "information". However, the same term "information" in the context of "information theory" becomes much more important, because without this term it is not clear at all which theory is referred to. Therefore, we should give the term "information" a lower weight in the context of "information retrieval", but a higher weight in the context of "information theory". This issue is specially important for Mandarin Chinese for which the syllable is a very useful indexing feature. In Mandarin many different homonym characters with different meanings may be pronounced as the same syllable. As a result, using a fixed weight for each syllable may not be helpful at all. But very often the exact character and its meaning for a syllable becomes clear once its context is given. So the context-dependent weighting scheme becomes important. It has been proposed to give different terms different decision thresholds [16], but our task here considers different weights for different terms in different queries.

For simplicity, we let the context of a term u be defined as the query it belongs to. So we change the weight $M(u)$ above into $M(u,q)$, i.e. the weight for a term u may be different in different query q. $\mathcal{M}(\mathcal{Q})$ is then the weight of all the terms existing in the query set $\mathcal{Q}$. It is not difficult to find $\mathcal{M}(\mathcal{Q}_{train})$, where $\mathcal{Q}_{train}$ is the training query set, but it will be difficult to adapt $\mathcal{M}(\mathcal{Q}_{train})$ into $\mathcal{M}(\mathcal{Q}_{test})$, where $\mathcal{Q}_{test}$ is the testing query set, because many queries in $\mathcal{Q}_{test}$ are not supposed to be in $\mathcal{Q}_{train}$.

3.1. Training Stage

Similar to (2), we wish to solve (5) to maximize the evaluation measure on the training data,

$$\hat{\mathcal{M}}(\mathcal{Q}_{train}) = \arg \max_{\mathcal{M}(\mathcal{Q}_{train})} \sum_{i=1}^{m} E(y_i, r_i(\mathcal{M}(\mathcal{Q}_{train}))). \quad (5)$$

Because each query has its own weight for its terms, changing weights for one query do not affect the weights for other queries. Also, as we mentioned in Section 2.1, all the training queries only contain two terms(two characters or two syllables). As a result, solving (5) is not difficult, and we do not need to use SVM-MAP. Since there are only two weights for the two terms to be estimated in a query, and the total number of training queries is not very large, we can find the two weights by simply exhaustively searching through all combinations. Note that scaling the weights by a constant do not change the ranking result, so for unique solution without losing the generality, we confine the weights to range between 0 and 1, and the sum of two weights in a query to be 1. So an exhaustive search through all combinations of the two weights for a query between 0 to 1 by some fixed step size is not difficult and the best combination of weights can be found for all terms in the training query set. For queries not in $\mathcal{Q}_{train}$, we simply set the two weights to be 0.5.

3.2. Testing Stage

We only have context-dependent weights for bi-grams of terms, but there are many queries having more than two terms. This can be handled by scaling the weights but requiring the ratio between the two weights for the two terms in all term bi-grams within the query to be the same as those trained. In other words, given a query of L terms $q = [u_1, u_2, ..., u_L]$, we simply scale the weights by requiring

$$\frac{M(u_{i+1}, q)}{M(u_i, q)} = \frac{M(u_{i+1}, u_i u_{i+1})}{M(u_i, u_i u_{i+1})}, i = 1, 2, ..., L-1, \quad (6)$$

because we assume the best ratio between the weights in a long query is the same as those in a bi-gram query. This is clearer with the following example. Consider a query "spoken document retrieval", it is a query with three terms, so it is not in $\mathcal{Q}_{train}$. However, as long as "spoken document" and "document retrieval" are in $\mathcal{Q}_{train}$, we can get the best weights respectively. Assume

$$\begin{aligned} M(spoken, \text{“}spoken\ document\text{”}) &= 0.7 \\ M(document, \text{“}spoken\ document\text{”}) &= 0.3 \\ M(document, \text{“}document\ retrieval\text{”}) &= 0.1 \\ M(retrieval, \text{“}document\ retrieval\text{”}) &= 0.9. \end{aligned}$$

So, from (6) we have

$$\begin{aligned} M(spoken, \text{“}spoken\ document\ retrieval\text{”}) &= 0.7 \\ M(document, \text{“}spoken\ document\ retrieval\text{”}) &= 0.3 \\ M(retrieval, \text{“}spoken\ document\ retrieval\text{”}) &= 2.7\ . \end{aligned}$$

4. EXPERIMENTAL RESULTS

4.1. Experimental Setting

The audio data used in our experiments is the Mandarin broadcast news stories collected daily from local stations in Taiwan from 2000 to 2001. It included 8 hours of training set and 45 minutes of testing set. The acoustic models used included a total of 151 intra-syllable right-context-dependent Initial-Final models for Mandarin syllables trained with the training set. The tri-gram language model was based on a 62K lexicon. The testing set is manually divided into 5034 segments, each viewed as a document for retrieval. From the lattices for these segments, we generated the corresponding subword-based PSPL using Chinese characters and Mandarin syllables. The character accuracy for the 5034 segments is 75.27% (under tri-gram one-pass decoding) and the syllable accuracy is 81.98%.

When assigning weights to character terms, 905 important characters were selected based on tf/idf scores evaluated from a text corpus, and a total of 906 characters were considered with 1 extra representing all other characters. When assigning weights to syllable terms, all the 399 syllables phonologically allowed were used. In the training stage, 1675 character bi-grams and 2474 syllable bi-grams with high tf/idf scores were automatically selected as the training query. For the testing stage, 158 text test queries were chosen manually from a set of automatically generated candidates, which consisted of two to nine characters and appeared at least five times in the 5034 segments. 38 of the 158 queries included out-of-vocabulary (OOV) words and were categorized as OOV queries, while the other 120 words were in-vocabulary (IV) queries. Only 55 testing queries out of the 158 had exactly the same patterns of character/syllable bi-grams as those in training queries.

All retrieval results are presented in terms of MAP, evaluated by the standard trec_eval package from the TREC evaluations. We evaluated the retrieval results using two kinds of terms, character and syllable, and we also evaluated the MAP for all testing queries ,and for IV or OOV testing queries only.

4.2. Proposed Context-independent Weighting

We tested the first proposed approach of context-independent weighting. We used the SVM^{map} developed by Yisong Yue and Thomas Finley [8] for SVM-MAP. The tradeoff parameter $\mathcal{C}$ is set empirically. The results are shown as the second bar in each group for characters and syllables respectively in of Figure 1, as compared to the baseline of using term frequencies (tf), actually the posterior probabilities, only. We see that the proposed context-independent weighting approach performed superior than the baseline approach using posterior probability only as weighs when using the characters as the term. But the situation was completely different when using the syllable as the term, for which MAP was degraded after the weighting. This is expected, because in Mandarin each syllable usually represents many different characters with different meanings, therefore giving each syllable a single fixed weight regardless of the context is simply inappropriate. This problem is much less serious for characters, which is why improved MAP was obtained when using character as the term. We also see the retrieval accuracy of using the character alone as the indexing feature was far more better than that using the syllable alone. This is natural because each syllable usually represents many different characters and is therefore much more confusing.

4.3. Proposed Context-dependent Weighting

The results for the second proposed approach of context-dependent weighting are shown in Figure 1 as the third bar in each group. We can see that using either character or syllable, MAP scores were consistently improved as compared with the baseline of posterior probability only and the first proposed approach of context-independent weighting. This is reasonable. Many Chinese characters also have different meanings depending on the context, although this happens less frequently as that for syllables. So context-dependent weighting for characters offered reasonable improvements for characters.

For syllables, on the other hand, almost all syllables are shared by different characters with different meanings in different context. So context-dependent weighting offered very significant improvements with respect to not only the baseline of posterior probabilities case, but completely solved the problem of performance degradation of context-dependent weighting.

Note that we do have a scheme to weight each term in a long query with more than two terms in the second approach of context-dependent weighting, although we are not sure if the weights obtained in that way is optimal. However, it was observed from the data that for most queries longer than two terms the MAP scores were actually improved when using the second approach of context-dependent weighting, as compared to the first proposed approach of context-independent weighting. Nevertheless, it also seemed true that more significant improvements were observed for short queries with two terms only than those for longer queries.

The above are the results for the two feature weighting schemes applied on characters or syllables separately. Of course, combination of them is another very interesting possibility. We combined scores obtained from character features with context-independent(CI) and context-dependent(CD) weights with those from syllable features. So there is a total of four combinations, character (CI or CD) plus syllable (CI or CD). The results are shown in Figure 2, compared with combination of baseline character and syllable features using only posterior probabilities. The combination ratio between syllable and character scores has been chosen empirically. We can obviously see that the combination of character and syllable features properly weighted significantly outperformed the baseline character and syllable features using only posterior probabilities. The results also pointed out that the combination of context-independent weighting on characters plus context-dependent weighting on syllables achieved the best performance among the four combination.

The result in Figure 2 is thought-provoking because context-dependent weighting did better than context-independent on both syllables and characters. But when we consider more carefully the characteristics of Chinese language, this may be reasonable. Many characters do have different meanings in different context, so context-dependent weighting works better for characters alone. But these context dependency also has two different types: some characters with different meanings in different context are also pronounced as different syllables, but some others with different meanings in different context are pronounced as the same syllable. We are not sure which type appears more frequently. But it is very possible that those two different types caused different degree of disturbance to the context-dependent weights of the syllables. The final result therefore indicated characters with context-independent weights combined with syllables with context-dependent weights were the best.

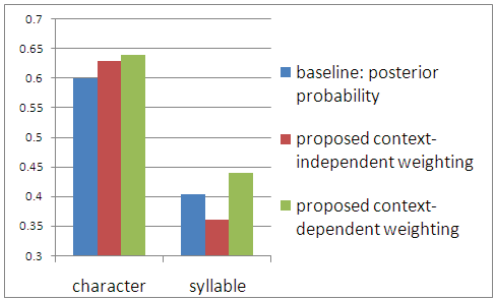

Fig. 1. MAP results of baseline of using posterior probability only, and the two proposed context independent/dependent weighting approaches, respectively for characters and syllables.

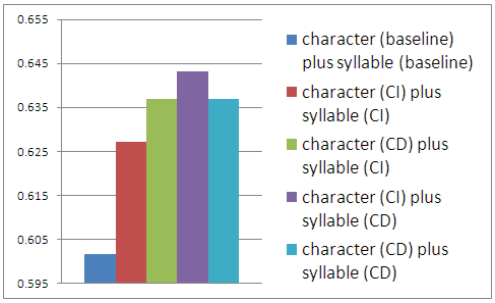

Fig. 2. MAP results of combinations of context-independent(CI) and context-dependent(CD) weighting approaches for syllables and characters.

5. CONCLUSION

In this paper, we propose to tune the weights to be applied on each indexing feature by optimizing the retrieval performance measure. Preliminary experiments using unigrams of Chinese characters and syllables over broadcast news in Mandarin Chinese indicate that both weighting approaches proposed here, context independent or dependent, were able to offer very promising performance under various conditions. Further work on words and N-grams considering proximity information over different corpora is currently under progress.

6. REFERENCES

[1] J. Tejedor, D. Wang, S. King, J. Frankel, and J. Colas, "A posterior probability-based system hybridisation and combination for spoken term detection," in *INTERSPEECH*, 2009.

[2] K. Iwata, K. Shinoda, and S. Furui, "Robust spoken term detection using combination of phone-based and word-based recognition," in *INTERSPEECH*, 2008.

[3] D. R. H. Miller, M. Kleber, C. lin Kao, and O. Kimball, "Rapid and accurate spoken term detection," in *INTERSPEECH*, 2007.

[4] J. S. Olsson, J. Wintrode, and M. Lee, "Fast unconstrained audio search in numerous human languages," in *ICASSP*, 2007.

[5] C. Chelba and A. Acero, "Position specific posterior lattices for indexing speech," in *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 2005.

[6] N. Bertoldi, R. Zens, and M. Federico, "Speech translation by confusion network decoding," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, 2007.

[7] P. Yu, Y. Shi, and F. Seide, "Approximate word-lattice indexing with text indexers: Time-anchored lattice expansion," in *ICASSP*, 2008.

[8] Y. Yue, T. Finley, F. Radlinski, and T. Joachims, "A support vector method for optimizing average precision," in *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 2007.

[9] C. H. Meng, H. Y. Lee, and L. S. Lee, "Improved lattice-based spoken document retrieval by directly learning from the evaluation measures," 2009.

[10] B. Matthews, U. Chaudhari, and B. Ramabhadran, "Fast audio search using vector space modelling," in *ASRU*, 2007.

[11] J. Mamou, D. Carmel, and R. Hoory, "Spoken document retrieval from call-center conversations," in *SIGIR*, 2006.

[12] V. T. Turunen and M. Kurimo, "Indexing confusion networks for morph-based spoken document retrieval," in *SIGIR*, 2007.

[13] J. S. Olsson, "Vocabulary independent discriminative term frequency estimation," in *Interspeech*, 2008.

[14] V. T. Turunen, "Reducing the effect of oov query words by using morph-based spoken document retrieval," in *Interspeech*, 2008.

[15] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, "Support vector machine learning for interdependent and structured output spaces," in *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, 2004.

[16] D. Wang, S. King, J. Frankel, and P. Bell, "Term-dependent confidence for out-of-vocabulary term detection," in *INTERSPEECH*, 2009.