A Language Independent Approach to Audio Search

Vikram Gupta[1], Jitendra Ajmera[2], Arun Kumar[3], Ashish Verma[2]

[1] Electrical Engineering Department, IIT Delhi, New Delhi, India
[2]IBM Research-India
[3]CARE, IIT Delhi, New Delhi, India

vikram.nov.14@gmail.com, jajmera1@in.ibm.com, arunkm@care.iitd.ernet.in, vashish@in.ibm.com

Abstract

In this paper, we propose an approach towards audio search where no language specific resources are required. This approach is most useful in those scenarios where no training data exists to create an automatic speech recognition (ASR) system for a language, e.g. in the case of most regional languages or dialects. In this approach, a Multilayer perceptron (MLP) is trained for a language where the training data exists, e.g. English. This MLP estimates a sequence of probability vectors for an audio segment, which is referred to as the *posteriorgram representation* for that segment. Components of the probability vector are posterior probabilities of English phonemes at any given frame of speech. Template matching technique is then used to compare the query-posteriorgram against the content-posteriorgram over the searchable audio-content. We present experiments in this paper to show that, even for other language like Hindi, the probabilities obtained from the neural network trained on English provide a characteristic representation for a word. A dynamic time warping algorithm with appropriate modifications is applied and encouraging P@N performance of *46.24%* for Hindi and *65.22%* for English for the task of audio search is reported while using the same MLP trained using English data in both the cases.

Index Terms: audio search, posteriorgram, MLP, dynamic time warping, language independent, KL divergence

1. Introduction

Recent years have witnessed an exponential growth in the amount of audio content generated and consumed every day by users in the form of audio books, news archives, call centres archives etc. The efficient use of audio content requires effective search techniques to retrieve relevant information from huge audio databases which turns *audio search* into a very interesting area of research. One of the interesting approaches towards handling the problem is to use large vocabulary continuous speech recognition systems (LVCSR) to convert the audio into text and subsequently perform text search on it [1,2]. However, LVCSR based systems require trained acoustic models for the target language, which may not be available for every language. LVCSR based systems also have limitations while dealing with out-of-vocabulary (OOV) words.

The problem related to OOV words can be approached by adopting techniques based on sub-word units like syllables and phones [3]. However, these approaches still require trained language specific acoustic models. Approaches using phonetic posteriorgram representation of the audio, followed by template matching algorithms have reported promising audio search performance [4,5] in a mono-lingual scenario.

Techniques exploiting acoustic models of a resource rich language as seed models and then adapting them for a new target language with minimal resources have been studied [6,7] too, however, the acoustic models developed are still language specific. The portability of tandem features (derived from phone and articulatory feature MLPs) languages without any retraining has also been investigated for ASR [8].With over 6000 languages in the world, approaches which can cater to multiple similar languages with no intervention may prove to be extremely effective.

In the work presented in this paper, we use a multi-layer perceptron (MLP) trained on English to generate a phonetic posteriorgram representation. This is done for both the query utterance as well as all the utterances in the searchable audio-content. An efficient dynamic time-warping algorithm [9,10] is then applied to align these two representations to find all putative occurrences of the query within the searchable audio-content. We present experiments on English as well as Hindi and show that without requiring any training material for Hindi, we can obtain reasonable search accuracy.

The novelty of this work lies in transferring a posteriorgram based audio search framework trained on the English language to the Hindi language without any tailoring. *The Dynamic Time Warping Algorithm* has been modified to favor diagonal path extensions and combat unconstrained endpoint warping problem efficiently. The modifications also make our warping algorithm computationally less expensive as compared to [4]. The performance of KL (Kullback-Leibler) divergence and dot product as the distance measures has also been analyzed.

This paper is organized hereafter as follows: Section 2 explains the posteriorgram representation and its generation using MLP. Section 3 presents the modified DTW search algorithm. Section 4 presents the experimental setup and the results, followed by conclusions in Section 5.

2. Posteriorgram representation

2.1 Training of MLP

The MLP has been trained on the WSJ1 corpus [12]. An utterance is sampled into a sequence of frames and the Mel Frequency Cepstral Coefficients (MFCC) along with the delta and delta-delta features are extracted from each frame. Time aligned phonetic transcription for each frame is generated using a trained recognizer. MFCC features (along with delta and delta-delta) of 9 successive frames along with the phonetic label for the middle frame are fed as input to a multi-layer perceptron (MLP) having one hidden layer and *m* output units corresponding to the *m* phonetic classes.

Based on the supervised input, the MLP tries to learn the optimum weights by using the *back propagation algorithm*. The *Quicknet tools* [11] developed at ICSI have been used to train the MLP.

2.2 Phonetic Posteriorgram

The MLP thus trained as above, is used to estimate a posterior probability vector given an MFCC vector. The posterior probability vector is an *m* dimensional vector where *m* equals the number of phonetic classes (fixed to 40 in our case, corresponding to the 40 English phones). A given utterance (query or audio content) can be converted into a sequence of probability vectors and this resulting representation over time is referred to as a posteriorgram. Figure 1 and Figure 2 show an example of a posteriorgram for audio snippets corresponding to "*South Korean Airliner*" and "*yahan se lagbhag*" ("*approximately from here"* in Hindi).

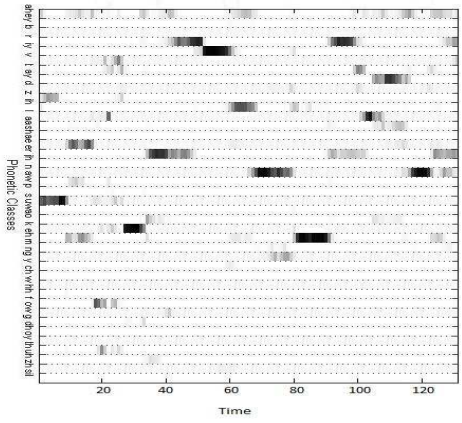

Figure 1: A posteriorgram for an English audio snippet "*South Korean Airliner*" with time on x-axis and phonetic class probability on y-axis obtained using English trained MLP.

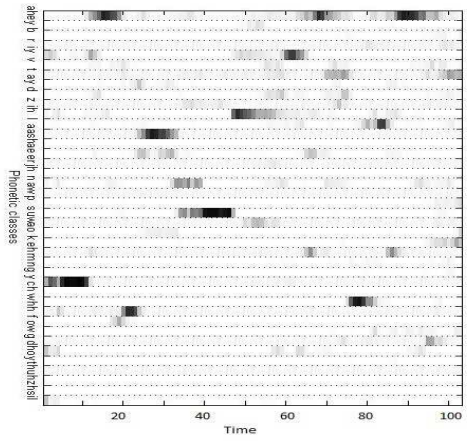

Figure 2: A posteriorgram for a Hindi audio snippet "*yahan se lagbhag*" with time on x-axis and phonetic class probability on y-axis obtained using the English trained MLP.

The entropy of the posteriorgram shown in Figure 1 comes out to be 0.0250 while for the posteriorgram in Figure 2 entropy is measured as 0.036 which shows that the probability distribution in the posteriorgram corresponding to the Hindi audio is more evenly distributed.

3. The search algorithm

Dynamic Time Warping (DTW) belongs to the class of non-parametric approaches where all the information present in the data is used to represent the data, without approximating any statistics for the data, as done in the case of parametric approaches. Given a reference and a test representation, the objective of the DTW is to find the best path that spans both the representations while minimizing a given cost function. DTW can efficiently handle speaking rate variability and durational differences that are always encountered in speech problems.

In the scenario presented in this paper, a direct implementation of DTW is undesirable considering that a query representation has to be matched against only a segment within the searchable audio-content. A direct implementation was reported in [4] but we found that it results in unmanageable computational and memory requirements. A modification to the DTW algorithm is therefore explored and is presented in the following sub-section.

3.1. Modified DTW algorithm

We use a modified *DTW Algorithm* [9,10] to efficiently score the best possible path corresponding to a query in each of the database utterance. An ***X*Y*** matrix is formed, where ***X*** and ***Y*** are respectively the number of frames in the query and in the utterance. Each cell of the matrix stores the accumulated normalized cost corresponding to the best possible path to that cell and discards all other costs. Thus, we need to visit each cell only once unlike in [4], where multiple visits to cells falling on overlapping paths, starting from adjacent starting points are needed, which makes our algorithm computationally less expensive. At the end, the normalized cost corresponding to the last frame of the query are compared to find out the cell with the least accumulated normalized cost. This cell becomes the ending point of the best possible path.

The process discussed above is done in the following way: First, a local cost between the i^{th} frame of the query and the j^{th} frame of the database utterance is computed as follows:

$$LC_{i,j} = \frac{D_{i,j}}{\underset{i=1\ to\ M}{max}\ (D_{i,j})} \quad (1)$$

M is the number of frames present in the query. $D_{i,j}$ is the distance measure between frame i of the query to the frame j of database utterance, which will be discussed in the next section.

As done in DTW, we propagate through a possible path and compute the accumulated cost associated with it. To allow comparison of these costs across paths having variable length, we normalize the accumulated cost by its length to result in a normalized distance, $NC_{i,j}$, as follows:

$$NC_{i,j} = min \frac{(NC_{k,l} * L_{k,l} + W * LS_{i,j})}{1 + L_{k,l}} \quad (2)$$

$$(k,l) \in \{(i-1,j-1), (i,j-1), (i-1,j)\}$$

$L_{k,l}$ is the length of the warped path till the predecessor.

The slope factor W penalizes the horizontal and vertical path extensions and thus discourages the paths with higher flatness or steepness which is undesirable as the speaking variations

are usually less. *W* is always set to 1 for diagonal path extensions.

3.2 Distance measures

The DTW algorithm requires a distance measure to calculate the dissimilarity between two frames as represented by the term D_{ij} in (1). In our experiments, we have focussed on two distance measures: Dot Product and KL divergence along with its variants. Suppose $\boldsymbol{q}$ and $\boldsymbol{u}$ represent the two *m* dimensional probability vectors corresponding to the two frames of *query* and *database utterance*, respectively.

The *asymmetric KL Divergence* is defined as:

$$D_{Asym}(\mathbf{q}||\mathbf{u}) = \sum_{i=1}^{i=m} \mathrm{q(i)} \log\left\{\frac{\mathrm{q(i)}}{\mathrm{u(i)}}\right\} \quad (3)$$

To study the effect of interchanging the role of query frame ($\boldsymbol{q}$) and database utterance frame ($\boldsymbol{u}$), we define *reverse asymmetric KL* as:

$$D_{Asym}(\mathbf{u}||\mathbf{q}) = \sum_{i=1}^{i=m} \mathrm{u(i)} \log\left\{\frac{\mathrm{u(i)}}{\mathrm{q(i)}}\right\} \quad (4)$$

We also perform experiment to evaluate the performance of *symmetric KL* defined as:

$$D_{Sym}(\mathbf{u}||\mathbf{q}) = \frac{1}{2}\left[\sum_{i=1}^{i=m} \mathrm{u(i)} \log\left\{\frac{\mathrm{u(i)}}{\mathrm{q(i)}}\right\} + \mathrm{q(i)} \log\left\{\frac{\mathrm{q(i)}}{\mathrm{u(i)}}\right\}\right] \quad (5)$$

Since the posteriorgram representations are probability distributions over the phonetic classes, KL Divergence should perform better as a distance measure than the dot product. However, to compare our approach with [5], we also studied the impact of using dot product on the accuracy. The dot product between a frame, $\boldsymbol{u}$ of the database utterance and $\boldsymbol{q}$ of the query, is defined as follows:

$$D(\mathbf{q},\mathbf{u}) = -\log\left\{\frac{q.u}{|q||u|}\right\} \quad (6)$$

4. Experiment and Results

4.1 MLP training

Acoustic models trained using HTK toolkit were used to generate the phone level transcriptions for 30,000 utterances (58 hours duration) from the WSJ1 (Wall Street Journal) [12] corpus. The MLP has one hidden layer with 1000 units and "*softmax*" output activation function. MFCC features (along with delta and delta-delta) corresponding to the 9 successive frames along with the phonetic label of the middle frame is used as supervised input to train the MLP, thereby also modeling the contextual information. 25000 audio utterances were used as training samples and 5000 utterances were used for cross validation.

Table 1 reports the effect of changing the number of neurons in the hidden layer on the frame accuracy. As the number of hidden neurons increases, the cross validation accuracy improves. However, the training and decoding time also increases.

For further experiments with English and Hindi databases, MLP with one hidden layer and 1000 hidden units with 351 input units corresponding to the 9 successive frames with 39 dimensional features each and 40 output states corresponding to the 40 English phonemes has been used.

Table 1. *Effect of the number of hidden neurons on accuracy*

Number of Hidden Neurons	Training Accuracy	Cross Validation Accuracy
500	74.92%	68.60%
1000	75.14%	69.91%
2000	75.50%	70.74%
3000	75.89%	71.74%

The audio database for experiments on English keywords consisted of 3000 audio utterances taken from WSJ0 corpus [13]. One instance of each of the 20 keywords was randomly clipped from the WSJ1 corpus.

Table 2. *Keywords along with their number of instances present in 3000 audio utterances from English database.*

Computer:12	Possible:10	Manager:15	Worker:20
Chemical:10	Standard:13	Domestic:12	Treasury:15
Partner:10	Important:10	Political:13	Women:18
Percentage:12	Protection:14	Revenue:9	Senate:11
Continental:13	Insurance:10	Agreement:14	Magazine:8

We report **P@10** (average precision for top 10 hits) and **P@N** (average precision for the top *N* hits, where *N* is the number of instances of the keyword in the database).

4.2 Effect of Slope Factor (*W*)

Table 3 reports the effect of changing the slope factor over the precision. A slope factor of 2 means that *W*=2 for horizontal and vertical extensions. *W* is always fixed to 1 for diagonal extensions.

Table 3. *Performance (in percentage) on English data with varying slope factors (W) and asymmetric KL divergence as the distance measure*

W=1		W=2		W=3		W=4	
P@10	P@N	P@10	P@N	P@10	P@N	P@10	P@N
63.19	57.15	68.81	62.61	71.81	65.22	71.32	64.25

W=1 means that no penalty has been applied. The average precision improves with the slope factor till the value 3 and decreases after that, so a slope factor of 3 has been used for further experimentation.

4.3 Effect of distance measures

We repeated the experiments by changing the distance measure keeping all other things fixed. Performance of KL divergence against dot product with *W=3* is shown in Table 4.

Table 4. *Performance of distance measures on English data*

Distance measure	P@10	P@N
Asymmetric KL	71.81%	65.22%
Reverse Asymmetric KL	64.69%	58.15%
Symmetric KL	68.94%	63.87%
Dot Product	71.19%	64.45%

We used the same MLP trained using English data to generate posteriorgrams representation for Hindi utterances. DTW with the optimum value of slope factor (W=3) and asymmetric KL divergence (as defined in equation 3) was used on Hindi data also. We used 1000 utterances spoken by 100 speakers along with the transcriptions for the experimentation. Out of these, 729 audio utterances from 73 speakers were used as the content to be searched and single instances of 20 Hindi keywords were extracted from the remaining 271 utterances as query terms (shown in Table 5).

Table 5. *Keywords along with their number of instances present in 729 audio utterances from Hindi database.*

Daravaje:3	Bartan:79	Kapade:14	Vyavastha:4
Rashtrapati:1	Ascharya:9	Vasiyat:3	Lalamani:4
Karate:14	Utpadan:4	Guruseva:2	Bharat:21
Karane:15	Matlab:6	Laghbhag:79	Dhobin:79
Dekhakar:5	Bhumi:13	Majabut:13	Bhajapa:8

Since the audio data for Hindi was limited, the average number of instances for keywords is less as compared to English. Many of the keywords have instances less than 10 because of which only P@N has been reported on Hindi data. Table 6 shows the performance of asymmetric KL divergence and Dot Product over Hindi data.

Table 6. *Performance of distance measures on Hindi data*

Distance Measures	**P@N (W=3)**
Dot Product	37.01%
Asymmetric KL divergence	46.24%

From the experiments and results we observe the following points:

(a) Improvements in performance (P@10 by 13.64% and P@N by 14.12% relative) are observed by applying a *slope factor* of 3, which can be explained by the fact, that the speaking variations are usually small making diagonal path extensions more probable.

(b) Use of a *modified DTW algorithm* makes this approach computationally less expensive as compared to [4] since each cell is visited only once and the score corresponding to only the best possible path to reach it is stored.

(c) Asymmetric *KL divergence* performs substantially better than dot product when applied to Hindi (*25% relative improvement in P@N*) since the distance corresponding to a highly probable component is weighted by its probability. The enhancement due to scaling becomes less effective in case of English, because probability distribution is peaky while in Hindi, it is much flatter. It is also important to note that reversing the role of query and database utterance degrades the performance.

(d) **P@N of 46.24%** is reported when the MLP trained using data from English language is applied to Hindi language without any tailoring, which is encouraging and comparable with the figures reported by [4,5] on English language itself.

(e) Also, our P@10 performance of 71.81% over English is comparable to the performance mentioned in [14] on English data. The approach in [14] uses a phoneme recognizer and compares various ways of matching phonetic lattices against ngram-based phonetic index. Although this comparison is not valid considering that we are not using exactly the same data, it presents the overall effectiveness of the system.

We also observed that a small subset of the Hindi keywords (having more number of phonemes than others): *Rashtrapati, Vasiyat, Ascharya, Lalamani, Vyavastha, Guruseva, Daravaje, Utpadan* and observed a P@N of 58% which is better than P@N of 38.40% for the remaining 12 keywords. This shows that the approach performs better for longer keywords.

5. Conclusions

This paper presents a language independent audio search technique where an MLP trained using data from English is applied to Hindi language to generate posterior probability vectors, without requiring any language specific resources. The use of an efficient modified DTW algorithm for searching incorporated with a slope penalty factor reports a **P@N of 46.24%** for Hindi and **P@N of 65.22%** for English, while using the English trained MLP in both the cases. The experiments also show that asymmetric KL divergence performs significantly better than the dot product, especially in the case of Hindi language.

Acknowledgements

The authors would like to thank Dr.K.Samudravijaya, TIFR and Dr.S.Lata, TIFR for providing Hindi speech database.

References

[1] D. Miller, *et al*, "Rapid and accurate spoken term detection," in *Proc. Interspeech*, Antwerp, Belgium, 2007.

[2] M.Saraclar and R.Sproat, "Lattice-based search for spoken utterance retrieval," in *Proc. HLT-NAACL*, Boston, 2004.

[3] K. Ng, "Subword-based approaches for spoken document retrieval," Ph.D. dissertation, Massachusetts Institute of Technology, 2000.

[4] Y.Zhang and James.R.Glass, "Unsupervised spoken keyword spotting via segmental DTW on Gaussian Posteriorgrams," in *Proc. ASRU*, 2009

[5] T. Hazen, W. Shen and C. White, "Query-by-example spoken term detection using phonetic posteriorgram templates," in *Proc. ASRU*, 2009

[6] Andrej Zgank et.al "Crosslingual transfer of source acoustic models to two different target languages", *COST278 and ISCA Tutorial and Research Workshop on Robustness Issues in Conversational Interaction* (Robust'04), Norwich (United Kingdom), August 2004.

[7] T.Schultz and A.Waibel "Adaptation of Pronunciation Dictionaries for Recognition of Unseen Languages," Workshop on Speech and Communication, SPECOM 1998, Russia.

[8] O.Cetin et.al, "Monolingual and crosslingual comparison of tandem features derived from articulatory and phone mlps," in *Proc. ASRU*, 2009

[9] J. Junkawitsch, L. Neubauer, H. Hoge, and G. Ruske, "A new keyword spotting algorithm with pre-calculated optimal thresholds," in ICSLP, Philadelphia, 1996, vol. 4, pp. 2067-2070.

[10] J.Ajmera and F.Metze, "Keyword spotting using durational entropy," in Proc. ICASSP 2007

[11] "http://www.icsi.berkeley.edu/Speech/qn.html"

[12] "http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC94S13A"

[13] "http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93S6A"

[14] W. Shen, C. White, and T. Hazen, "A comparison of query-by-example methods for spoken term detection," pp 2143-2146, INTERSPEECH, 2009.