CUNY Systems for the Query-by-Example Search on Speech Task at MediaEval 2015

Min Ma[1], Andrew Rosenberg[2]
[1]Department of Computer Science, CUNY Graduate Center, USA
[2]Department of Computer Science, Queens College (CUNY), USA
mma@gradcenter.cuny.edu, andrew@cs.qc.cuny.edu

ABSTRACT
This paper describes two query-by-example systems developed by Speech Lab, Queens College (CUNY). Our systems aimed to respond with quick search results from the selected reference files. Three phonetic recognizers (Czech, Hungarian and Russian) were utilized to get phoneme sequences of both query and reference speech files. Each query sequence were compared with all the reference sequences using both global and local aligners. In the first system, we predicted the most probable reference files based on the sequence alignment results; In the second system, we pruned out the subsequences from the reference sequences that yielded best local symbolic alignments, then 39-dimension MFCC features were extracted for both query and the subsequences. Both the two systems employed an optimized DTW, and obtained C_{nxe} of 0.9989 and 1.0674 on the test data respectively.

1. INTRODUCTION
The primary goal of query-by-example search is to detect the occurrence of query audio in a unlabeled reference speech corpus. This year, this task emphasized the probabilistic decisions of the query occurrence, ignoring the exact start and end times of hit regions in the reference audio. Moreover, three different types of query searches were used in this year's evaluation in addition to exact matches of query audio: 1) word level re-orderings, 2) small filler content and 3) conversational queries in context. Detailed description can be found in [1]. Therefore, we targeted comprehensive search systems which are compatible with the queries of different types, and capable to output both strict and approximate matches. Our systems took advantage of phonetic symbolic sequences as well as MFCC features, and employed a DTW algorithm with novel optimizations. Note that all the queries were manually recorded with the acoustic context, in order to avoid any problems when cutting the queries from a longer sentence. However, this context might interfere the retrieval performance. Thus, we filtered out the exact spoken queries as the ones searched in our systems, using the timing boundaries of the relevant words provided.

2. PHONETIC TOKENIZER
Unlike resource rich languages, for low-resource languages, automatic speech recognition still remains challenging due to the lack of adequate training data. Without sufficient data for language modeling, or reliable pronunciation modeling, we seek to decode the audio to a phoneme sequence. We utilized three phonetic tokenizers (Czech, Hungarian and Russian) released by Brno University of Technology (BUT) to generate the phoneme transcripts[2]. All BUT recognizers used the split temporal context network structure. All queries and reference audios were decoded by the three tokenizers and non-phonetic symbols (*pau, int, spk*) were removed from final sequences.

MediaEval 2015 Workshop, September 14-15, 2015, Wurzen, Germany

3. SEQUENCE ALIGNMENT
Considering various acoustic and voice conditions of audio and the error rate of phonetic tokenizers, phonetic sequences from the query and the actual hit region in the reference audio may not always be *exactly* the same, but should be *similar*. Thus we employed three aligners to match phoneme sequences: 1) strict global aligner, 2) (fuzzy) global aligner and 3) local aligner, where 1) was designed for exact matches, while 2) and 3) for inexact matches. Every aligner used Levenshtein edit distance and backtracking to find the optimal matching parts of each sequence pair, yielding a matching score and a similarity percentage. We performed alignment for all the sequences decoded by different tokenizers, thus obtained 9 matching scores and 9 similarity percentages for each sequence pair.

4. SYSTEM 1: SMO+ISAX
In our first system, we investigated how to use global and local symbolic similarities to identify the best possible reference files for DTW to search.

Binary Classifier: We aggregated all the scores and percentages computed from Section 3 for each query-reference (*q-ref*) pair, then assigned "CORR" ("correct", if the query occurred in the reference audio) or "FA" ("false alarm") for training queries. The data set showed strong between-class imbalance, where "FA" samples was 1,365 times more than "CORR" instances. Since most classification algorithms do not perform optimally when class examples are extremely imbalanced, we applied clustering centroid undersampling [3], a technique proven effective to address such issue, to our training data. We then trained a support vector machine using sequential minimal optimization (SMO)[4] and evaluated using four-fold cross validation on training data. Another SMO model was trained over all the training set and tested on test queries to make predictions of "CORR"/"FA". Limited by time, for each query, we narrowed our search scope of DTW to top 50 reference files that obtained high-

Table 1: An algorithm converting phoneme sequence $\mathcal{T}$ to time series $\mathcal{S}$($\mathcal{A}$ stands for phoneme alphabet)

```
Always set $s_1 = 0$;
For $\mathcal{T} = t_1, t_2, ..., t_N$, where $t_i \in \mathcal{A} = \{l_1, l_2, ..., l_M\}$ :
    $s_{i+1} = s_i + (\frac{M}{2} - j + 1)$ if $t_i = l_j$ and $j \leq \frac{M}{2}$;
    $s_{i+1} = s_i - (j - \frac{M}{2})$ if $t_i = l_j$ and $j > \frac{M}{2}$.
```

est prediction confidences.

Sequence Mapping: DTW is an excellent search algorithm, especially on small data sets [5]. Therefore, after using the edit distance to get the rough decisions (*i.e.* shrink the scope of candidates), we applied DTW to make more accurate decisions as to whether a spoken query occurs in the candidate audio segment. We utilized **i**ndexable iSAX[6], a method that uses the symbolic words to internally organize and index the data, and supports DTW by building bounding envelopes around the query time series. Following [6], we mapped the phoneme sequence using the approach shown in Table 1. The phoneme vocabulary was constructed by all the phonemes from query and reference sequences. To simplify the mapping algorithm, shown in Table 1, a "null" symbol was inserted at the end of the alphabet to ensure the total number is even. In this way, we converted phoneme sequences to integer time series, in order to apply the DTW described in Section 6.

5. SYSTEM 2: SUBSEQ+MFCC

In our second system, phoneme sequence alignments were used to determine the subsequences of reference files that best matched the query sequence. We then applied DTW over the subsequence and query pair (*q-subref*) by comparing their MFCC features, a type of acoustic features that has been widely used for the task.

Subsequence Detection: We made an approximate estimation of start and end time points of the subsequence that was likely to be an occurrence of query, only for the reference files that achieved a high average similarity from local aligners. More specifically, we selected the top 100 *q-subref* pairs for each query and extracted the hypothesized matched regions from reference audio segments. For example, suppose we have a query sequence *"u i k"*, then only the region that were transcribed as *"u i k"* would be searched. Therefore, using the symbol matching looking, we only need make DTW comparisons between the query sequence and a smaller local region of the reference segment.

MFCC Feature Extraction: We extracted 39-dimension MFCC features for both query and subsequence speech files using OpenSMILE [7]. The features were produced from 25 ms audio frames (sampled at a rate of 10 ms), by computing 13 MFCC (0-12) from 26 Mel-frequency bands and applying a cepstral liftering filter with a weight parameter of 22, with 13 delta and 13 acceleration coecients appended to the MFCC. Finally, we made mean normalization of all the features with respect to the full input sequence.

6. OPTIMIZED DTW

Four optimization techniques proposed in [5] can greatly accelerate the DTW algorithm: 1) early abandoning z-normalization, 2) reordering early abandoning, 3) reversing the query/data role in LB_{Keogh} and 4) cascading lower bounds. UCRsuite [8] implements the optimized DTW algorithm. In the first system, optimized DTW was applied to the iSAX-converted time series; in the second system, it was employed to compare each dimension of MFCC features, then the 39 scores were aggregated to compute their average. Per-query normalization was performed, using the formula $z = \frac{x-\mu}{\sigma}$ where μ is the mean of raw matching scores for the query, and σ is their standard deviation.

Table 2: System Performances on all the queries (System 1 is primary system)

System	$actC_{nxe}$	$minC_{nxe}$	ATWV	MTWV
SMO+iSAX-dev	0.9988	0.9872	0.0011	0.0067
SMO+iSAX-eval	0.9989	0.9870	0.0006	0.0010
Subseq+MFCC-dev	1.0658	0.9823	-3.9820	0.0123
Subseq+MFCC-eval	1.0674	0.9853	-4.0205	0.0006

7. EXPERIMENT RESULTS

Results obtained by our systems on both dev queries and eval queries are shown in Table 2. System **1** performed better than System **2**, in both C_{nxe} and $ATWV$ metrics. Smaller C_{nxe} indicates System **1** is a more informative system of the two, which might be caused by the fact that System **1** chose the candidate files based on both (strict) global and local similarities, while System **2** selected the subsequences that either matched the entire query(*e.g. "white horse"*) most, or matched part of the query(*e.g. "horse"*) most, which might fail to capture some inexact matches as Type 2 and Type 3 queries and misrecognized files that just partially matched the query. Therefore, System **2** performed worse, even though it introduced acoustic information from MFCC and generated a response faster. Besides, basing decisions on the phoneme sequences made the accuracy of phoneme recognizers impact the performance of both systems. To run the experiments, we used a computer with 60CPUs (2.8 GHz, 120 cores), 64GB RAM and 1TB hard drive. Most of the computation cost of our systems was caused by the alignment and DTW phases. Approximately, the Indexing Speed Factor was 0.607, Searching Speed Factor was 0.307 per sec and per language, and Peak Memory was 1.903 GB for the primary system.

8. CONCLUSIONS

We briefly summarize our two systems as part of the MediaEval QUESST 2015 Shared Tasks along with their evaluation results. Our first system involved (strict) global/local similarities of symbolic phoneme sequences, therefore was compatible to both the exact query search and approximate search tasks. Additionally, a classifier-based selection method of candidate reference files were explored. Our second system used local similarity as measurement to filter out the most similar subsequences to query sequence, thus reduced the computational burden of DTW comparison. In our systems, optimized DTW algorithms were employed to compare either iSAX-ed phoneme sequences or 39-dimension MFCC features to determine if a query appears within a reference sample. In the future, more acoustic features and selection approaches should be further investigated.

9. REFERENCES

[1] I. Szöke, L.-J. Rodriguez-Fuentes, A. Buzo, X. Anguera, F. Metze, J. Proenca, M. Lojka, and X. Xiong, "Query by example search on speech at MediaEval 2015," in *Working Notes Proceedings of the MediaEval 2015 Workshop, Sept. 14-15, 2015, Wurzen, Germany, 2015.*

[2] P. Schwarz, "Phoneme Recognizer Based on Long Temporal Context," in *http://www.fit.vutbr.cz/~schwarzp/publi/thesis.pdf ,PhD Thesis, Brno University of Technology, 2008.*

[3] M. M. Rahman and D. Davis, "Cluster Based Under-Sampling for Unbalanced Cardiovascular Data," in *Proceedings of the World Congress on Engineering*, vol. 3, 2013.

[4] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy, "Improvements to Platt's SMO Algorithm for SVM Classifier Design," in *Neural Computation*, vol. 13, no. 3, 2001, pp. 637–649.

[5] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and Mining Trillions of Time Series Subsequences under Dynamic Time Warping," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2012, pp. 262–270.

[6] J. Shieh and E. Keogh, "iSAX: Indexing and Mining Terabyte Sized Time Series," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2008, pp. 623–631.

[7] F. Eyben, F. Weninger, F. Groß, and B. Schuller, "Recent Developments in openSMILE, the Munich Open-Source Multimedia Feature Extractor," in *Proceedings of the 21st ACM international conference on Multimedia.* ACM, 2013, pp. 835–838.

[8] "The UCR Suite," in *www.cs.ucr.edu/~eamonn/UCRsuite.html*.