ELiRF at MediaEval 2015: Query by Example Search on Speech Task (QUESST)

Sergio Laguna, Marcos Calvo, Lluís-F. Hurtado, Emilio Sanchis
Departament de Sistemes Informàtics i Computació
Universitat Politècnica de València
Camí de Vera s/n, 46020, València, Spain
{slaguna, mcalvo, lhurtado, esanchis}@dsic.upv.es

ABSTRACT

In this paper, we present the systems that the Natural Language Engineering and Pattern Recognition group (ELiRF) has submitted to the MediaEval 2015 Query by Example Search on Speech Task. All of them are based on a Subsequence Dynamic Time Warping algorithm. The systems use information from outside the task (*low-resources systems*).

1. INTRODUCTION

In this paper, we present the two systems that we have submitted to the MediaEval 2015 Query by Example Search on Speech Task [7]. Both systems are based on a Subsequence Dynamic Time Warping (S-DTW) algorithm [2]. However, they differ in the way the minimization process is performed. In the first system the minimization is computed directly on the accumulated distances, while in the second system the average distances are considered. In the following sections, we will explain how the feature vectors are computed for each system, the differences in the search algorithm, the scoring system and the results obtained in this evaluation.

2. OVERVIEW OF THE SYSTEMS

Both of our systems use the same philosophy. The first step was to preprocess all the audio files, both spoken documents and queries, using a phoneme recognizer. This way a sequence of feature vectors based on posteriorgrams was obtained as a representation of each audio file [3, 5]. Then, we took each possible pair (*document, query*) and run a S-DTW algorithm on them. This provided the bounds of a possible detection of the query within the document, and a score for this detection. After that, a decision-making module established a threshold based on the scores of all the possible detections. This was necessary in order to only accept the detections with the highest confidences. Finally, a fusion module used the scores obtained from three systems based on three different languages and mixed them in order to achieve better results.

3. PREPROCESSING

We used the phoneme recognizer developed at the Brno University of Technology [6] and adopted the three available systems for 8 KHz audio: Czech, Hungarian and Russian.

MediaEval 2015 Workshop, Sept. 14-15, 2015, Wurzen, Germany

The systems were externally trained using the SpeechDat corpus.

With this phoneme recognizer a vector representation of the audio files was built. The recognizer uses internally a HMM architecture with a Neural Network representation for each phoneme, and it outputs the posterior probability for each of the three states of each unit (45 for Czech, 61 for Hungarian and 52 for Russian) at every audio frame. Three of the units do not represent actual phonemes, but they represent noise or silence. These posterior probabilities conform the feature vectors for each language, also called posteriorgrams [8]. The probabilities obtained for different languages are not mixed.

At this point, the query files were cut to remove the context. Also, the information from the non-phonetic units was used to remove leading and trailing noises or silences.

4. SEARCH ALGORITHM

The search algorithm we used is based on Dynamic Programming (DP). In particular, we used S-DTW, which is a DP technique for comparing two sequences of objects. In our case, one of the sequences corresponded to feature vectors of one of the audio documents, and the other one represented a query. The S-DTW method found multiple local alignments of the query within an audio document, by allowing it to start at any position of the audio document. Equation 1 shows the generic formulation of S-DTW:

$$M(i,j) = \begin{cases} +\infty & i < 0 \\ +\infty & j < 0 \\ 0 & j = 0 \\ \min\limits_{\forall (x,y) \in S} M(i-x, j-y) + D(A_i, B_j) & j \geq 1 \end{cases} \tag{1}$$

where M is the DP matrix; S is the set of allowed movements, represented as pairs (x, y) of horizontal and vertical increments; A_i, B_j are the objects representing the i-th and j-th positions of their respective sequences; and D is a function that computes some distance or dissimilarity between two objects. In our case, we used the minus logarithm of the dot product as the distance function.

In our systems the set of allowed movements S was {(1,2), (1,1), (2,1)}. This set of movements guarantees that the size of any detection will be between 0.5 and 2 times the size of the query. This search procedure was run separately for each of the three languages considered, shaping this way the first system, which will be referred to as SDTW.

The second system (SDTW-avg) is based on a variation on the minimization process which takes into account the number of steps the S-DTW algorithm does [4]. This way, a local minimization is performed on the average distances, by keeping track of the total distance and the number of steps. This method was also run separately for each of the three languages.

5. SCORING AND FUSION

Once the search algorithm had found the best alignment for each possible pair (*document, query*), the distance values were used to determine the scores. The score must indicate how likely it is that a query appears in a document. For this reason, the minus distance was used as score (a smaller distance indicates then a better score). For each query a different threshold based on the n-best matches among all the documents is used. The selected pairs go then through a score normalization process, so the scores regarding each query have zero-mean and unit-variance. A default score is assigned to the pairs that are not selected.

Previous works such as [1] have shown that a proper combination of systems leads to better results. In our case a combination of the systems that used each of the three languages is performed. However, our combination is not as elaborated as the one from that work. Our combination system was to keep for each pair (*document, query*) the maximum score among the three considered languages. Then, a threshold is calculated for the hard decision (YES/NO), used for the Term Weighted Value metric. This threshold was a close value to the one calculated by the scoring tool for the Maximum Term Weighted Value with the development set. This fusion by language was performed in both of our systems.

6. EXPERIMENTS AND RESULTS

We performed several preliminary experiments in order to find the best configuration for our systems. For example, we evaluated different distance functions for the search algorithm. One of them was the Kullback-Leibler divergence, since the audios were represented as sequences of probability vectors. We also tried the cosine distance. Finally, we used the dot product, since it provided the best results for the development set.

For this MediaEval 2015 Query by Example Search on Speech Evaluation, we submitted one run for each of the systems described above. The results we obtained are shown in Tables 1 and 2. The measure to be optimized for this Evaluation was the Normalized Cross Entropy Score (C_{nxe}). However, other measures such as the Maximum and the Actual Term Weighted Values (MTWV and ATWV, respectively) were considered as secondary metrics. In these tables SDTW stands for the system where the minimization was based on the distance, and SDTW-avg stands for the system where the minimization was based on the average distance.

Table 1: Results obtained for the development set.

System	C_{nxe}	C_{nxe}^{min}	ATWV	MTWV
SDTW-avg	1.0651	0.8677	0.1446	0.1543
SDTW	1.0701	0.8702	0.1404	0.1493

Table 2: Results obtained for the test set.

System	C_{nxe}	C_{nxe}^{min}	ATWV	MTWV
SDTW-avg	1.0731	0.8751	0.1125	0.1181
SDTW	1.1879	0.9338	0.0449	0.0581

The figures that are shown in the tables reveal better results if the minimization is performed using the average distance rather than if the total distance is used.

The results shown in Tables 3 and 4 are separated by query type. Our approach obtains better results with query type T1 (exact match), since it does not naturally consider the word level reorderings or possible filler items from query types T2 (with small variations and word level reorderings) and T3 (split with fillers).

Table 3: Results for the test set (SDTW-avg).

Query type	C_{nxe}	C_{nxe}^{min}	ATWV	MTWV
T1	0.9167	0.7870	0.1978	0.2043
T2	1.1276	0.9052	0.0755	0.0836
T3	1.1381	0.8959	0.0801	0.0939

Table 4: Results for the test set (SDTW).

Query type	C_{nxe}	C_{nxe}^{min}	ATWV	MTWV
T1	1.0641	0.8675	0.1006	0.1166
T2	1.2316	0.9524	0.0241	0.0385
T3	1.2413	0.9526	0.0111	0.0370

These systems use our own multi-thread implementation of the S-DTW algorithm. We used a PC with an Intel® Xeon® @ 3.70 GHz with 12 threads and 64 GB of RAM on a Linux operating system. At the preprocessing stage, we achieved an *indexing speed factor* of 0.40, and our memory peak was lower than 0.1 GB. At the search stage, our *searching speed factor* was 3.11 and the memory peak was above 13 GB. Thus, our *processing load* for both systems was 37.29.

7. CONCLUSIONS

In this paper, we have presented the systems we have submitted to the MediaEval 2015 Query by Example Search on Speech Evaluation, as well as the results obtained. This was a very challenging task in which not only exact occurrences of the queries, but also with lexical variations, word level reorderings or queries that were split and had fillers in between had to be found. Our approach offers better results where an exact match is searched, as they are more natural to our search algorithm. We have also tried a variation on the standard S-DTW algorithm which considers the average distance instead of the raw sum of distances. This variation led to better results on the test set.

As future work, we would like to improve our system in order to better handle the word reorderings, as well as queries that are not represented as a consecutive piece of audio.

8. ACKNOWLEDGEMENTS

This work is funded by the Spanish Government and the E.U. under contract TIN2014-54288-C4-3-R and FPU Grant AP2010-4193.

9. REFERENCES

[1] A. Abad, L. J. Rodríguez-Fuentes, M. Penagarikano, A. Varona, and G. Bordel. On the calibration and fusion of heterogeneous spoken term detection systems. In *INTERSPEECH*, pages 20–24, 2013.

[2] X. Anguera and M. Ferrarons. Memory efficient subsequence DTW for Query-by-Example spoken term detection. In *2013 IEEE International Conference on Multimedia and Expo*. IEEE, 2013.

[3] T. J. Hazen, W. Shen, and C. White. Query-by-example spoken term detection using phonetic posteriorgram templates. In *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*, pages 421–426. IEEE, 2009.

[4] A. Muscariello, G. Gravier, and F. Bimbot. Audio keyword extraction by unsupervised word discovery. In *INTERSPEECH 2009: 10th Annual Conference of the International Speech Communication Association*, 2009.

[5] A. Muscariello, G. Gravier, and F. Bimbot. Zero-resource audio-only spoken term detection based on a combination of template matching techniques. In *INTERSPEECH 2011: 12th Annual Conference of the International Speech Communication Association*, 2011.

[6] P. Schwarz. Phoneme Recognition based on Long Temporal Context, PhD Thesis. *Brno University of Technology*, 2009.

[7] I. Szöke, L. J. Rodriguez-Fuentes, A. Buzo, X. Anguera, F. Metze, J. Proença, M. Lojka, and X. Xiong. Query by Example Search on Speech at Mediaeval 2015. In *MediaEval 2015 Workshop*, September 14-15, 2015.

[8] Y. Zhang and J. R. Glass. Unsupervised spoken keyword spotting via segmental dtw on gaussian posteriorgrams. In *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*, pages 398–403. IEEE, 2009.