DOUBLE-LAYER NEIGHBORHOOD GRAPH BASED SIMILARITY SEARCH FOR FAST QUERY-BY-EXAMPLE SPOKEN TERM DETECTION

Kazuo Aoyama, Atsunori Ogawa, Takashi Hattori, and Takaaki Hori

NTT Communication Science Laboratories, NTT Corporation
2-4, Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0237, Japan

ABSTRACT

This paper presents a novel double-layer neighborhood graph index for acceleration of similarity search that accomplishes fast query-by-example spoken term detection (STD). When a query segment is given, our proposed STD method finds similar segments to the query from an utterance data set by efficient similarity search that traverses the double-layer neighborhood graph (*DLG*) with a low computational cost. The segment is a sequence of Gaussian mixture model posteriorgram frames and corresponds to a vertex in the *DLG*. A dissimilarity between vertices is measured by dynamic time warping. The *DLG* consists of two distinct degree-reduced k-nearest neighbor graphs in a base and an upper layer. The base layer's graph has all the vertices in the data set while the upper layer's graph includes only representatives extracted from the vertices in the base layer. By way of analogy, search in the *DLG* resembles driving on *general roads* and *express highways* appropriately for travel-time saving. Experimental results on the MIT lecture corpus demonstrate that the proposed method achieves CPU time reduction by 40% and more than 60% compared to the most recent method and the ordinary graph-based method, keeping almost the same precision.

***Index Terms*—** Spoken term detection, Query-by-example search, Search index, Neighborhood graph, Dynamic time warping

1. INTRODUCTION

Increasing use of various and large speech data cultivates greater interest in spoken term detection (STD). A popular approach is to convert the speech data into linguistic representation in advance by automatic speech recognition (ASR), and then to execute search by a text query term [1]. This approach is useful for tasks which are well-resourced in terms of ASR model construction. In contrast, the demand for approaches in low-resource situations has increased recently, where ASR is not available [2]. Under such situations, query-by-example STD is a useful candidate that does not require any type of prior linguistic knowledge such as transcriptions, language models, or pronunciation dictionaries [3, 4].

Unfortunately, query-by-example STD has drawbacks of low performance in accuracy and speed in spite of its potential availability. The low accuracy is attributed to the fact that STD must rely only on the acoustic information without the linguistic knowledge. To solve the accuracy problem, many efforts have been continued [4–7]. The low speed comes from the fact that an indexing and a search algorithm are still under development. For the high speed performance, a template matching approach based on dynamic time warping (DTW) has been proposed [8–13]. In particular, the methods reported in [8, 9, 12, 13] share the common feature and dissimilarity; a Gaussian mixture model (GMM) posteriorgram as a feature of a time frame, and as a dissimilarity between two posteriorgrams, the negative logarithm of the probability that the two posteriorgrams are generated from the same distribution [14].

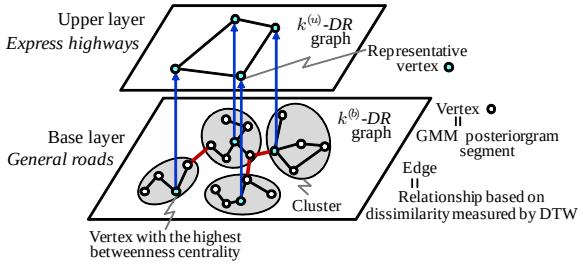


Fig. 1. A diagram of a double-layer neighborhood graph (*DLG*)

We focus on the improvement of the graph-based similarity search method (*GSS*) in [12], which uses neighborhood graph index. A vertex and an edge in the graph correspond to a GMM posteriorgram segment and the relationship between a pair of segments, which is measured by DTW, respectively. Once a graph index is constructed, *GSS* can be performed very fast, which is advantageous when search tasks for distinct queries are performed many times in the same data set.

In this paper, we present a novel double-layer neighborhood graph (*DLG*) index for *GSS* acceleration. Figure 1 shows a diagram of the *DLG*. The proposed STD method finds similar segments to a given query segment from an utterance data set by traversing a *DLG* with a simple search algorithm. The *DLG* consists of two distinct degree-reduced k-nearest neighbor (k-*DR*) graphs in a base and an upper layer. A $k^{(b)}$-*DR* graph in the base layer is constructed from all the vertices in the data set while a $k^{(u)}$-*DR* graph in the upper layer is independently constructed by using only representatives of the vertices in the base layer. Intuitively, we can understand the two distinct graphs in the base and the upper layer as *general roads* and *express highways* for travel-time saving, respectively. We demonstrate in experimental results on the MIT lecture corpus [15] that the proposed *DLG* method achieves CPU time reduction by 40% and more than 60% compared to the recent method *HGSS* in [13] and the ordinary *GSS* in [12], respectively, keeping almost the same precision.

2. RELATED WORK

We review three topics: STD by DTW, similarity search using a k-nearest neighbor graph family as an index, and centrality in graphs.

2.1. Spoken Term Detection Using Dynamic Time Warping

We can categorize fast STD by a template matching approach based on DTW into two types. One is an exact method that reduces the number of exact DTW-score calculations by using a DTW-score lower bound obtained just when a query segment is given, i.e., *online* [3, 8, 9]. The other is an approximate method that is charac-

terized by the use of an index built from an utterance data set *off-line* before search [10–13]. A typical *on-line* exact method is the STD using a DTW-score lower bound calculated with a low computational cost in [8], which we call *LB*. Furthermore, by applying a piecewise aggregation approximation technique to *LB*, the speed performance was improved [9]. In the approximate methods, there are some methods using different types of an index; a hash table generated by locality-sensitive hashing (*LSH*) [10], a k-means tree [11], and a k-*DR* graph [12, 13]. Note that the parameter k in the k-means algorithm denotes the number of centroids in clusters, and differs from graph structural parameter k in the k-*DR* graph.

2.2. Similarity Search Using a Neighborhood Graph Index

A k-nearest neighbor (k-*NN*) graph has been studied as a search index [16–20]. The k-*NN* graph has an edge between a pair of vertices x and y if x is among the k closest vertices to y or vice versa. Note that the term "close" means a concept measured by not only a distance but also dissimilarity. A k-*NN* graph has useful properties for search [16], which resemble those in small-world networks [21, 22], both a tendency of like to associate with like and a very small average shortest path length. These properties allow a search algorithm to reach the vertex closest to a given query vertex from an initial vertex chosen at random with a few steps. In particular, a k-*DR* graph, which is a member of k-*NN* graph family and is improved for achieving better search performance, has been applied to a variety of data sets: large-scale documents [16], images [23], GMMs with a Kullback-Leibler divergence as a dissimilarity [24], and GMM posteriorgram sequences [12, 13].

2.3. Centrality in Graphs

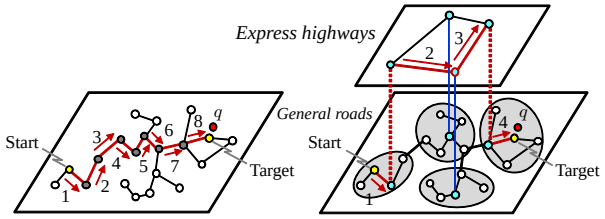
A proposed double-layer neighborhood graph (*DLG*) needs representatives of all the vertices in the base layer for constructing the upper layer. We first prepare clusters, i.e., subgraphs in the base layer in the same manner as that in [13]. We next extract a representative vertex from each subgraph by using *centrality*. Centrality is one of structural properties that characterize a graph (or network) [25]. Various types of centrality have been proposed, for instance, degree centrality, closeness centrality, graph centrality, betweenness centrality, and eigenvector centrality. In these, the betweenness centrality of a vertex in a graph is popular and widely used. The betweenness centrality is basically an appearance frequency of a vertex on shortest paths between all the pairs of vertices in the graph. Moreover, a fast algorithm for calculating the betweenness centrality has been reported in [26]. We adopt the betweenness centrality as a measure to select a representative of a subgraph, as detailed in Section 5.1.

3. PROBLEM FORMULATION

We deal with query-by-example STD as the following problem identical to that in [12] (For details to [12]). Suppose that a set of GMM posteriorgram sequences produced from an utterance data set, multiple query GMM posteriorgram segments q_i $(i = 1, 2, \cdots, m)$ produced from spoken instances for an identical query keyword, a definition of a normalized DTW score between q_i and x (x is a GMM posteriorgram segment), and the number of resultant utterances T are given. Then, efficiently find the T-best utterances based on a fusion score of the multiple query GMM posteriorgram segments.

4. GRAPH-BASED SIMILARITY SEARCH

We solve the foregoing problem by a graph-based similarity search method (*GSS*) characterized by the following three points; preconstructing multiple k-*DR* graphs as index candidates for a query segment with any length, selecting an appropriate index from the candidates when a query is given, and performing search in the k-*DR* graph with a multi-start greedy search (*MSGS*) algorithm followed by a breadth-first search (*BFS*) algorithm. We briefly review an ordinary and a hierarchical *GSS* [12, 13], in terms of the common k-*DR* graph and each search algorithm, to shed light on our proposed method.



(a) Ordinary *GSS* (b) *DLG* based *GSS*

Fig. 2. Conceptual diagram of greedy search paths in (a) the k-*DR* graph and (b) the *DLG* when the greedy search algorithm finds "Target" closest to the query q from "Start". The path length 4 in the *DLG* is smaller than 8 in the k-*DR* graph.

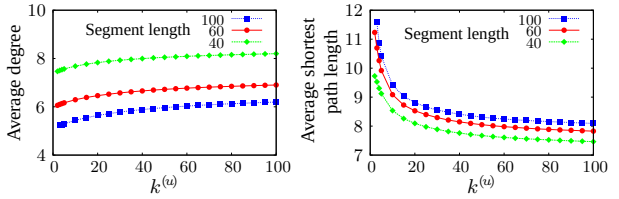
4.1. Ordinary *GSS*

A k-*DR* graph is a subgraph of a k-*NN* graph, and has fewer edges (a smaller degree) [16]. The degree reduction is effective for fast graph search. This is because a computational cost of a graph search algorithm with a greedy manner correlates with the product of an average degree and an average shortest path length, and the average shortest path length of the k-*NN* graph is already small to some extent for its small-world property [16]. Furthermore, to maintain navigation function of a small-world graph [27], the k-*DR* graph is designed so that a greedy search algorithm can reach each vertex from at least its k-th closest vertex as the k-*NN* graph.

The ordinary *GSS* employs a combination of a multi-start greedy search (*MSGS*) and a breadth-first search (*BFS*) to find the Top-T vertices closest to a query vertex [12]. The *MSGS* algorithm starting from multiple initial vertices finds the closest vertex to the query at a high search success rate, and collects a tentative set of the top-T vertices (a tentative top-T set) on the paths. The successive *BFS* algorithm determines the top-T set by updating the tentative top-T set as follows. The *BFS* algorithm repeatedly adopts a vertex at which the *MSGS* algorithm terminated in the tentative top-T as a root (initial) vertex in ascending order of its dissimilarity to the query. Each iteration terminates if the tentative top-T set is not updated at a current depth.

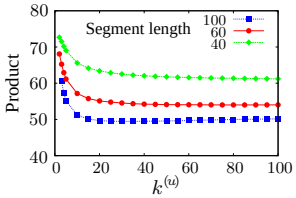
4.2. Hierarchical *GSS*

The Hierarchical *GSS* (*HGSS*) achieves faster STD than the ordinary *GSS* by dynamically reducing degrees *on-line* [13]. When a k-*DR* graph is constructed referring to a K-*NN* list ($K \geq k$), hierarchical clustering for the vertices in the graph is simultaneously performed by using the same list. Each cluster is regarded as a supernode in the hierarchical structure and given a cluster identification number (CIDN). Note that the edges in the k-*DR* graph are maintained with no change. The *HGSS* executes a combination of a *MSGS* and a *BFS* algorithm in the k-*DR* graph, which differs from the search algorithm in the ordinary *GSS* in added edge-selection function (or vertex-selection function) using the CIDNs. The *MSGS* algorithm selects vertices whose DTW-scores have to be calculated from all adjacent vertices based on their CIDNs, i.e., it reduces the number of DTW-score calculations. The *BFS* algorithm terminates early by using the CIDN. Thus the *HGSS* limits the search space, resulting in the reduction of the computational cost.



(a) Average degree

(b) Average shortest path length



(c) Product of average degree and average shortest path length

Fig. 3. Structural properties of double-layer neighborhood graphs (*DLG*) with segment lengths of 40, 60, and 100 along $k^{(u)}$: (a) average degree, (b) average shortest path length, and (c) product of average degree and average shortest path length, which correlates with a computational cost in search.

5. PROPOSED GSS METHOD USING A DLG INDEX

We tackle the acceleration of similarity search for fast query-by-example STD. A key is a novel search index, a double-layer neighborhood graph (*DLG*). Figure 2 shows a conceptual diagram of greedy paths in the k-*DR* graph and the *DLG* when the greedy search algorithm starting from "Start" vertex finds "Target" vertex closest to the given query q. In Fig. 2, the *DLG* decreases the path length by half, compared to the k-*DR* graph. The *DLG* can reduce an average shortest path length by controlling the structural parameter k in the two distinct layers. Note that the proposed method using the *DLG* differs from the recent *HGSS* in both the index structure and the search algorithm. In simple terms, the proposed method transfers a part of computational costs during search in the *HGSS* to a cost for constructing its index structure, *DLG*.

5.1. Representative Vertices

The *DLG* consists of two distinct k-*DR* graphs; a $k^{(b)}$-*DR* in the base layer and a $k^{(u)}$-*DR* graph in the upper layer. The $k^{(b)}$-*DR* graph is ordinarily constructed from all the vertices (segments) in a given utterance data set while the $k^{(u)}$-*DR* graph is independently constructed from only representative vertices in the base layer. Each representative is a vertex with the highest vertex (node) betweenness centrality in a subgraph in the $k^{(b)}$-*DR* graph. The subgraph is defined as a graph consisting of the following vertex set and edge set. The vertex set is a set of vertices that are given the same CIDN by the hierarchical clustering described in Section 4.2. The edge set is obtained by removing between-cluster edges and keeping within-cluster edges in the $k^{(b)}$-*DR* graph. A search algorithm starting from the representative seems to reach any vertex in the subgraph with a few steps with high probability because the representative most frequently appears in shortest paths of all the pair vertices in the subgraph.

5.2. Double-Layer Neighborhood Graph

A *DLG* diagram is shown in Fig. 1. To determine structural parameters $k^{(b)}$ and $k^{(u)}$ of the graphs, we verified statistical properties on k of an ordinary k-*DR* graph for the MIT lecture corpus (detailed in Section 6.1); an average degree, an average shortest path length, and their product that relates to a computational cost in a graph search. Note that the k value affects both computational cost in search and search accuracy. The search accuracy is in particular controlled by both the k value and search parameter L, which is the number of starting vertices in a *MSGS* algorithm [23]. In this paper, we adopted $k^{(b)} = 20$ based on our observational results.

The *DLG* is built by a fusion of the $k^{(b)}$-*DR* and the $k^{(u)}$-*DR* graph. The graph fusion is simply expressed as follows. Let $\Gamma = (V, E)$ denote the *DLG*, where V and E are sets of vertices and edges, respectively. Let $\Gamma_{k^{(b)}} = (V, E^{(b)})$ and $\Gamma_{k^{(u)}} = (V^{(u)}, E^{(u)})$ denote the $k^{(b)}$-*DR* and the $k^{(u)}$-*DR* graph, where $V^{(u)}$ is the representative set ($V^{(u)} \subset V$), respectively. Then $\Gamma = \Gamma_{k^{(b)}} \cup \Gamma_{k^{(u)}}$, $E = E^{(b)} \cup E^{(u)}$.

Figure 3 shows the statistical properties of *DLG* candidates along the $k^{(u)}$ value: (a) average degree, (b) average shortest path length, and (c) the product of them, when different segment lengths of 40, 60, and 100 are used. Regardless of the segment lengths, the average degree slightly increased and the average shortest path length drastically decreased in the small $k^{(u)}$ region. The product of the average degree and the average shortest path length was almost stable in the $k^{(u)}$ region of more than 20. From the standpoint of the search cost, the smaller product is desired. We adopted $k^{(u)} = 30$ for search performance evaluation.

5.3. Search in a Double-Layer Neighborhood Graph

We apply the simple search algorithm in [12] to the *DLG*, which is the combination of the *MSGS* and the *BFS* algorithm. The key of the acceleration of similarity search is in the graph structure as the index rather than the search algorithm. Owing to the simple search algorithm, we can suppress the computational load in search other than the DTW-score calculation since the search algorithm does not distinguish the edges $E^{(b)}$ and $E^{(u)}$ in the distinct layers, namely, it does not need the edge selection as *HGSS* does. Until now, we focus on the graph structure for the acceleration without consideration of the search accuracy. In the *GSS* methods, we can control the accuracy by search parameter L, the number of starting vertices in the *MSGS* algorithm reported in [23]. Considering the trade-off between the speed-up and the accuracy, we determined $L = 40$.

6. EXPERIMENTS

6.1. Experimental Settings

For search performance evaluation, we used the MIT lecture corpus [15] for an utterance data set, which consisted of 54,581 utterances for a training set and 3,000 utterances for a test set. A set of 13-dimensional Mel-Frequency Cepstral Coefficients (MFCCs) was extracted from the utterance set. The sampling rate was 16 kHz, and the frame size and the frame shift were 25 ms and 10 ms, respectively. A GMM with 50 mixture components was estimated from the set of the MFCC vectors in the training set. A 50-dimensional posteriorgram for each frame in the utterance set was produced from the GMM. For STD, we chose 20 keywords and picked up 10 distinct spoken instances for each keyword as the query from the training set. This was done to set up the similar conditions to those in [3, 11–13]. We employed a DTW score normalized by a query segment length as the dissimilarity, and obtained the 100-best utterances which contained the most similar 100 distinct segments to the query. In the DTW, the warping path was limited within the bandwidth $R = 6$ as in [3]. The 100-best-utterance lists, where each list consisted of 100-best utterances for each query, were merged into a single list for

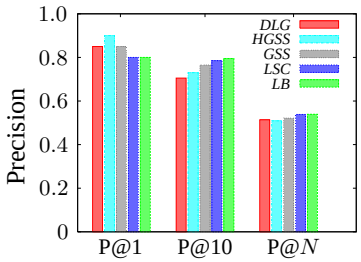


Fig. 4. Average precision of proposed method *DLG* and others; *HGSS*, *GSS*, *LSC*, and *LB*. The average precisions of all the methods were nearly equal regarding P@N.

each keyword. The utterances in the merged single list were sorted in ascending order of the fusion score calculated likewise to previous studies; it was done in the same manner as [12, 13]. Then the fusion parameter was set as $\alpha = 0.5$ equally to [3].

We compared the proposed method *DLG* with four existing methods: *HGSS* [13], *GSS* [12], *LSC* [12], and *LB* [8]. The comparison was carried out regarding two search performance criteria; the accuracy and the on-line computational costs, i.e., search speed. The accuracy was evaluated by the average precisions of the sets of top-X utterances to the ground truth for all the keywords; $X = 1, 10, N$, where N denotes the number of the correct utterances. The search speed was measured by three ways: the number of the DTW score calculations, the number of local dissimilarity calculations, and CPU time. We performed the experiments on a computer system equipped with an Intel Xeon E7-4870 2.4 GHz.

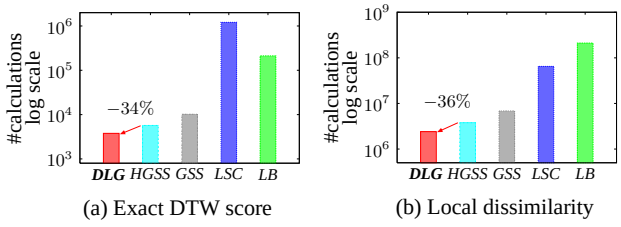
The *HGSS* and the *GSS* method employed 10 distinct k-*DR* graphs with different segment lengths $(20, \cdots, 100, 110)$. The graph structural parameter k was fixed as $k = 100$. The search parameter L for the *MSGS* algorithm was set as $L = 30$. The *DLG* method also employed 10 distinct $k^{(b)}$-*DR* graphs with the same segment lengths as those of the *HGSS* and the *GSS*. In each cluster in the $k^{(b)}$-*DR* graph, which was the same as that in the *HGSS*, a vertex with the highest betweenness centrality was extracted as the representative. In the upper layer, the $k^{(u)}$-*DR* graph was constructed from the representative vertices. The graph structural parameters were set as $k^{(b)} = 20$ and $k^{(u)} = 30$. The number of initial vertices L for the *MSGS* algorithm was fixed as $L = 40$.

6.2. Search Performance Evaluation

The proposed *DLG* accomplished the speed-up of query-by-example STD and also at nearly the same accuracy as the other methods. Figure 4 shows the average precisions of the sets of top-X utterances, which are denoted by P@X, $X = 1, 10, N$, of the proposed *DLG*, *HGSS*, *GSS*, *LSC*, and *LB*.

Figures 5(a) and (b) show the average numbers of DTW-score calculations and local-dissimilarity calculations with *logarithmic scale*, respectively. Compared to the *HGSS*, the proposed *DLG* reduced the numbers of the DTW-score calculations and the local dissimilarity calculations by 34% and 36%, respectively. These numbers were much smaller than those of *LSC* and *LB* by one order to two orders of magnitude. One reason of the reduction is appropriate selection of the structural parameters, i.e., $k^{(b)} = 20$ and $k^{(u)} = 30$ in the *DLG*. The parameter values satisfied both of low average degree and small average shortest path length of the *DLG*, whose product strongly relates to the number of the calculations.

Figure 6 shows the CPU time required by the five methods with



(a) Exact DTW score (b) Local dissimilarity

Fig. 5. Average number of calculations in terms of (a) exact DTW score and (b) local dissimilarity.

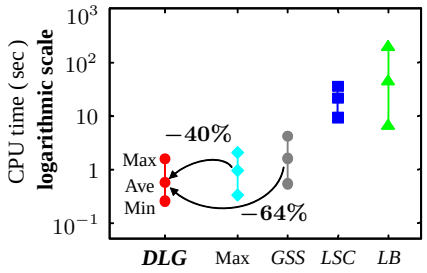


Fig. 6. CPU time of proposed method *DLG* and others; *HGSS*, *GSS*, *LSC*, and *LB*. *DLG* reduced the average CPU time by 40% and 64% compared to *HGSS* and *GSS*, respectively.

logarithmic scale. The three points of each method in Fig. 6, which are denoted by Max, Ave, and Min, correspond to the maximum, the average, and the minimum CPU time of all the spoken queries. The *DLG* reduced the average CPU time (Ave) by 40% than the *HGSS* that was the best method of the existing four, and by more than 64% than the *GSS* that was the baseline of the graph-based search. The reduction rate was higher than those of the numbers of the DTW-score and the local-dissimilarity calculations. This is because the search algorithm *on-line* in the *HGSS* selects vertices referring to its cluster identification number to limit search space, i.e., the *HGSS* incurs the additional computational costs, comparing to the simple search algorithm in the *DLG*. Besides, the *DLG* method operated in a much shorter average CPU time than the *LSC* and the *LB*. The rates of the required CPU time to those of the *LSC* and the *LB* were only $(27/1000)$ and $(13/1000)$, respectively.

7. CONCLUSION AND FUTURE WORK

We presented the novel graph-based similarity search method for fast query-by-example spoken term detection (STD). A newly introduced double-layer neighborhood graph (*DLG*) index allows the search method to quickly find similar segments to a given query segment from an utterance data set by traversing the *DLG*. The search in the *DLG* intuitively resembles driving to a destination by appropriately using *general roads* and *express highways* for travel-time saving, which correspond to the k-*DR* graphs in the base and the upper layer, respectively. Compared to the recent *HGSS*, the proposed *DLG* method reduced CPU time by around 40%, keeping almost the same precision. The acceleration of search by the *DLG* is attributed to its graph structure and the simple graph search algorithm *on-line*.

A search method using a layered neighborhood graph as an index has a lot of room to improve, for instance, a cluster definition and how to make clusters, extraction of cluster representatives, and selection of graph structures in the layers.

8. REFERENCES

[1] C. Chelba, T. J. Hazen, and M. Saraçlar, "Retrieval and browsing of spoken content," *IEEE Signal Process. Mag.*, vol. 25, no. 3, pp. 39–49, May 2008.

[2] J. Tejedor, M. Fapšo, I. Szöke, J. H. Černocký, and F. Grézl, "Comparison of methods for language-dependent and language-independent query-by-example spoken term detection," *ACM Trans. Inform. Syst.*, vol. 30, no. 3, August 2012.

[3] Y. Zhang and J. R. Glass, "Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams," in *Proc. Int. Workshop on Acoustic Speech Recognition & Understanding*, 2009, pp. 398–403.

[4] C.-A. Chan, C.-T. Chung, Y.-H. Kuo, and L.-S. Lee, "Toward unsupervised model-based spoken term detection with spoken queries without annotated data," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.* IEEE, May 2013, pp. 8550–8554.

[5] S. Soldo, M. Magimai.-Doss, J. Pinto, and H. Bourlard, "Posterior features for template-based ASR," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.* IEEE, May 2011, pp. 4864–4867.

[6] M. Huijbregts, M. McLaren, and D. van Leeuven, "Unsupervised acoustic sub-word unit detection for query-by-example spoken term detection," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.* IEEE, May 2011, pp. 4436–4439.

[7] H. Wang, T. Lee, C.-C. Leung, B. Ma, and H. Li, "Using parallel tokenizers with DTW matrix combination for low-resource spoken term detection," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.* IEEE, May 2013, pp. 8545–8549.

[8] Y. Zhang and J. R. Glass, "An inner-product lower-bound estimate for dynamic time warping," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.* IEEE, 2011, pp. 5660–5663.

[9] Y. Zhang and J. R. Glass, "A piecewise aggregate approximation lower-bound estimate for posteriorgram-based dynamic time warping," in *Proc. Interspeech*, 2011, pp. 1909–1912.

[10] A. Jansen and B. V. Durme, "Indexing raw acoustic features for scalable zero resource search," in *Proc. Interspeech*, 2012.

[11] G. Mantena and X. Anguera, "Speed improvements to information-retrieval-based dynamic time warping using heiarchical k-means clustering," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.* IEEE, May 2013, pp. 8515–8519.

[12] K. Aoyama, A. Ogawa, T. Hattori, T. Hori, and A. Nakamura, "Graph index based query-by-example search on a large speech data set," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.* IEEE, May 2013, pp. 8520–8524.

[13] K. Aoyama, A. Ogawa, T. Hattori, T. Hori, and A. Nakamura, "Zero-resource spoken term detection using hierarchical graph-based similarity search," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.* IEEE, May 2014, pp. 7093–7097.

[14] T. J. Hazen, W. Shen, and C. White, "Query-by-example spoken term detection using phonetic posteriorgram templates," in *Proc. Int. Workshop on Acoustic Speech Recognition & Understanding*, 2009, pp. 421–426.

[15] J. Glass, T. J. Hazen, L. Hetherington, and C.Wang, "Analysis and processing of lecture audio data: Preliminary investigations," in *Proc. HLT-NAACL*, 2004, pp. 9–12.

[16] K. Aoyama, K. Saito, T. Yamada, and N. Ueda, "Fast similarity search in small-world networks," in *Complex Networks: Int. Workshop on Complex Networks*. R. Menezes et al., Ed., 2009, pp. 185–196, Springer.

[17] T. B. Sebastian and B. B. Kimia, "Metric-based shape retrieval in large databases," in *Proc. Int. Conf. Pattern Recognition*, 2002, vol. 3, pp. 291–296.

[18] J. Sakagaito and T. Wada, "Nearest first traversing graph for simultaneous object tracking and recognition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, June 2007, pp. 1–7.

[19] K. Hajebi, Y. Abbasi-Yadkori, H. Shahbazi, and H. Zong, "Fast approximate nearest-neighbor search with k-nearest neighbor graph," in *Proc. Int. Joint Conf. Artificial Intelligence*, 2011, pp. 1312–1317.

[20] J.Wang and S. Li, "Query-driven iterated neighborhood graph search for large scale indexing," in *Proc. ACM Multimedia*, October 2012, pp. 179–188.

[21] D. J.Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, pp. 440–442, June 1998.

[22] J. Kleinberg, "The small-world phenomenon: An algorithmic perspective," in *Proc. ACM Symp. Theory of Computing*. SIGACT, May 2000, pp. 163–170.

[23] K. Aoyama, K. Saito, H. Sawada, and N. Ueda, "Fast approximate similarity search based on degree-reduced neighborhood graph," in *Proc. ACM SIGKDDD Conf. Knowledge Discovery and Data Mining*, August 2011, pp. 1055–1063.

[24] K. Aoyama, S. Watanabe, H. Sawada, Y. Minami, N. Ueda, and K. Saito, "Fast similarity search on a large speech data set with neighborhood graph indexing," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.* IEEE, March 2010, pp. 5358–5361.

[25] L. C. Freeman, "Centrality in social networks: conceptual clarification," *Social Networks*, vol. 1, pp. 215–239, 1978.

[26] U. Brandes, "A faster algorithm for betweenness centrality," *Journal of Mathematical Sociology*, vol. 25, no. 2, pp. 163–177, 2001.

[27] J. M. Kleinberg, "Navigation in a small world," *Nature*, vol. 406, pp. 845, August 2000.