UNIVERSITY OF JOENSUU
COMPUTER SCIENCE AND STATISTICS
DISSERTATIONS 22

Ville Hautamäki

Improving Pattern Recognition Methods for Speaker Recognition

Academic dissertation

To be presented, with the permission of the Faculty of Science of the University of Joensuu, for public criticism in the Louhela Auditorium of the Science Park, Länsikatu 15, Joensuu, on October 3th 2008, at 12 o'clock noon.

UNIVERSITY OF JOENSUU
2008

Supervisor Professor Pasi Fränti
Department of Computer Science and Statistics
University of Joensuu
Joensuu, FINLAND

Reviewers Professor Martti Juhola
Department of Computer Sciences
University of Tampere
Tampere, FINLAND

Professor Olli Nevalainen
Department of Information Technology
University of Turku
Turku, FINLAND

Opponent Professor Pekka Kilpeläinen
Department of Computer Science
University of Kuopio
Kuopio, FINLAND

ISBN 978-952-219-159-5 (printed)
ISBN 978-952-219-160-1 (PDF)
ISSN 1796-8100 (printed)
ISSN 1796-8119 (PDF)
Computing Reviews (1998) Classification: I.2.7, I.5.1, I.5.4, I.5.3, G.1.6, G.2.2

Joensuun yliopistopaino

Joensuu 2008

Improving Pattern Recognition Methods for Speaker Recognition

Ville Hautamäki

Department of Computer Science and Statistics
University of Joensuu
P.O.Box 111, FIN-80101 Joensuu, FINLAND
villeh@cs.joensuu.fi

University of Joensuu, Computer Science and Statistics, Dissertations 22
Joensuu, 2008, 126 pages

Abstract

AUTOMATIC speaker recognition is a very active area of research. The goal of speaker recognition is to either verify, based on voice only, that the user is who he claims to be or to identify unknown person from the voice sample. However, given the potential of the speaker recognition technology itself, it is still not widely used in commercial applications. The reason is that the speaker recognition is still an immature technology, where baseline technology is not very robust to various mismatches between training and testing conditions.

Currently, improvements to speaker recognition methodology are aimed either to pattern recognition techniques or signal processing and feature extraction procedures. In this thesis, the focus is mainly in the pattern recognition part.

Cluster analysis is the basic technology utilized in the speaker modeling. The first three publications improve the clustering results by speeding up the agglomerative clustering algorithm, developing an outlier removal technique, and creating a robust variant to the cluster centroid estimation.

A vector quantization based maximum *a posteriori* speaker model adaptation is also formulated. The adaptation strategy increases the accuracy of the vector quantization system to the same level as the state-of-the-art GMM-UBM system, while being 20 times faster in speaker training.

Mismatch compensation in speaker recogntion can be done for example by techniques from factor analysis. These recent techniques require external speech data, where statistical techniques will estimate the nuisance attributes. In this thesis, we propose a novel speaker matching and modeling method. The method is based on

graph matching and does not use any external training data.

Finally, we also propose a new voice activity detector for speaker recognition. We also systematically try out several speaker matching and feature extraction methods for the long-term average spectrum feature.

Keywords: Text-independent speaker recognition, vector quantization, cluster analysis, outlier detection, affine invariant matching, maximum a posteriori adaptation, unsupervised learning

Acknowledgements

I would like to express my sincere thanks to my supervisor Professor Pasi Fränti. I am especially grateful on the extremely successful collaborative work achieved with Dr. Tomi Kinnunen and Dr. Ismo Kärkkäinen. Other co-authors of the papers included in this thesis also deserve thanks, they are Dr. Olli Virmajoki, Svetlana Cherednichenko, Marko Tuononen, Juhani Saastamoinen and Tuija Niemi-Laitinen. Special thanks also goes for the proof reading done by Lomanzi Sakala and Pekka Nykänen.

I would also like to thank Professor Olli Nevalainen and Professor Martti Juhola for their effort on the review of this thesis. I am especially grateful for the helpful language editing provided by the above mentioned reviewers.

I would like to thank for the financial support East Finland Graduate School in Computer Science and Engineering (ECSE) during the years 2007-2008. The work on the thesis was also supported by the National Technology Agency of Finland (TEKES) as the four year project New Methods and Applications of Speech Technology (PUMS), and by all the government organizations and private companies participating in this project.

My parents and grand-parents have supported me tremendously during the long years of acquiring an education, for this and everything else they are greatly thanked for. Finally, I am forever indebted to my dear wife Rosa and our twin boys Diego and Samuel, they are the love of my life.

Joensuu, Tuesday, 9 September 2008

Ville Hautamäki

Let your vision be world-embracing
Bahá'u'lláh (1817–1892)

List of original publications

P1. P. Fränti, O. Virmajoki and V. Hautamäki, "Fast agglomerative clustering using a k-nearest neighbor graph", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 28, No. 11, pp. 1875–1881, November 2006.

P2. V. Hautamäki, I. Kärkkäinen and P. Fränti, "Outlier detection using k-nearest neighbour graph", in *Proceedings of the 17th International Conference on Pattern Recognition* (ICPR 2004), Vol. 3, pp. 430–433, Cambridge, UK, August 2004.

P3. V. Hautamäki, S. Cherednichenko, I. Kärkkäinen, T. Kinnunen and P. Fränti, "Improving k-means by outlier removal", in *Proceedings of the 14th Scandinavian Conference on Image Analysis* (SCIA 2005), pp. 978–987, Joensuu, Finland, June 2005.

P4. V. Hautamäki, T. Kinnunen, I. Kärkkäinen, J. Saastamoinen, M. Tuononen and P. Fränti, "Maximum *a posteriori* adaptation of the centroid model for speaker verification", *IEEE Signal Processing Letters*, Vol. 15, pp. 162–165, 2008.

P5. V. Hautamäki, T. Kinnunen and P. Fränti, "Text-independent speaker recognition using graph matching", *Pattern Recognition Letters*, Vol. 29, No. 9, pp. 1427–1432, July 2008.

P6. T. Kinnunen, V. Hautamäki and P. Fränti, "On the use of long-term average spectrum in automatic speaker recognition", in *Proceedings of the 5th International Symposium on Chinese Spoken Language Processing* (ISCSLP 2006), Vol. 2, pp. 559–567, Singapore, December 2006.

P7. V. Hautamäki, M. Tuononen, T. Niemi-Laitinen and P. Fränti, "Improving speaker verification by periodicity based voice activity detection", in *Proceedings of the 12th International Conference on Speech and Computer* (SPECOM 2007), Vol. 2, pp. 645–650, Moscow, October 2007.

Contents

Chapter 1

Introduction

BIOMETRIC person recognition from voice is a highly active research field [11]. The voice biometric has some advantages over more common biometric technologies, such as *fingerprint*, *iris* and *DNA* recognition. A person can be authenticated by his voice without him noticing it, as in border control usage scenario, where voice biometric system can run in the background while an officer is interviewing the passenger. Voice biometric system can also perform *realtime-recognition* [101, 113], for example, in call center operations, a user can be recognized while he is interacting with the system. The change of a user in the middle of the call can be detected and also more speech material can be collected during the whole call. *Face recognition* possesses similar properties but it is restricted to the operating situations where there is visual line of sight to the user.

Unfortunately, speech biometric is also inherently more difficult than more traditional means of person authentication using physical characteristics. Speech is a highly variable phenomenon. Variability is mostly due to the following aspects: (1) *speaker himself* (mood and health differences), (2) *technical conditions* (changes in environmental acoustics, transmission line and microphones, microphone setups) (3) *linguistic factors* (speech content, language, dialect and situation variations) [98]. The challenge in designing an automatic speaker recognition system is to make a system robust against the above mentioned variability and still retaining a good speaker discrimination ability.

An automatic speaker recognition system consists of the following components: voice activity detector (VAD), feature extraction, speaker modeling, pattern matching and speaker database. An overall system diagram is presented in Fig. 1.1. The speaker recognition task implemented in a complete system, is either *verification* or *identification*. In verification task, system is given an uknown speech sample and claimed speaker identity, the task is to decide whether unknown speech sample was

uttered by the claimed speaker. In the identification task, system will give speaker label to the unknown speech sample. Finally, the decision logic of the system returns the best scoring speaker or decision that the speaker is unknown.

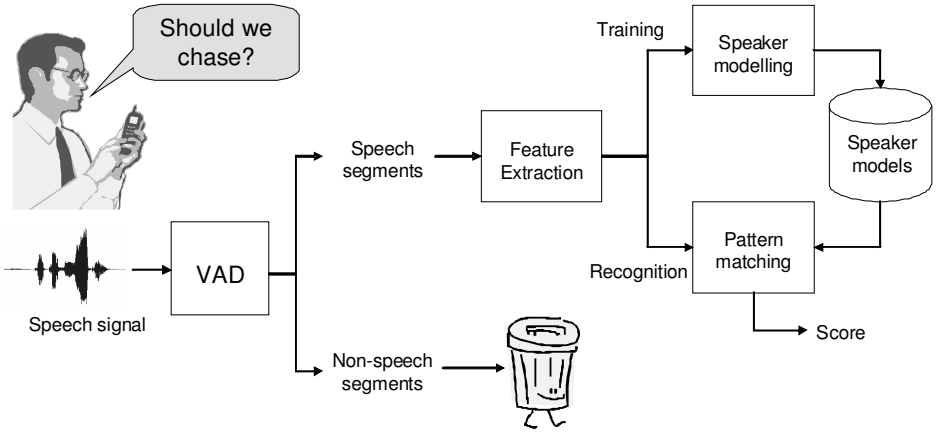


Figure 1.1: Structure of a speaker recognition system.

Training a new speaker to the speaker database and matching an unknown audio extract to the claimed model are basic operations in all speaker recognition systems. Typically, feature extraction produces a so called "feature cloud", which needs to be modeled somehow. Statistical techniques are commonly used in the modeling part of the system and data clustering is an essential tool therein.

The rest of the thesis is organized as follows. In Chapter 2, clustering is considered as an optimization problem, its computational complexity is analyzed and algorithmic solutions are reviewed. Gaussian mixture models and outlier detection methods are also discussed in the same chapter. Chapter 3 reviews the speaker recognition literature and shows how the clustering is applied in the speaker modeling. Summary of original research papers is given in Chapter 4 and the main results are summarized in Chapter 5. Finally, conclusions are drawn and future work outlined in Chapter 6. The original research papers are attached at the end of the thesis.

Chapter 2

Clustering

ONE can state that the intuitive goal of clustering is to find *natural clusters* from the input data without using any additional information [83, 185]. In machine learning, this type of learning problem is also known as *unsupervised learning* in contrast to *supervised learning* where additional class labels are provided with the data. The lack of additional information means that in order to be able to solve the *clustering problem*, one has to assume a clustering model. The algorithmic problem can be stated as; given a data set, find a set of model parameters that minimize (or maximize) a given objective function. If the data set includes class labels, the clustering algorithm can be objectively evaluated by checking for each pair of data vectors, whether they have been assigned to the same cluster or not [149]. Unfortunately, there is no way to know whether the results obtained with labeled training data will generalize to the unsupervised case.

Clustering is performed on the input data set (or *training set*) $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$ of N vectors in $\mathbb{R}^D$. The task is to partition X into k disjoint subsets (or *partitions*) S_i such that $\bigcup_{i=1}^{k} S_i = X$ and $S_i \cap S_j = \emptyset$ for all $j \neq i$. In the above definitions, each observation belongs to only one cluster. It is sometimes called *crisp* or *hard* clustering to distinguish from *soft* clustering definitions where an observation belongs to certain clusters by some membership degrees. There are number of soft clustering definitions including for example *fuzzy* [110], *probabilistic* [37] and *possibilistic* [109] clusterings.

Intuitively, each cluster should be internally as homogeneous as possible and, on the other hand, between-cluster similarity should be minimized [185]. Clustering, as defined above, does not include any *hierarchy* of clusters. Some clustering algorithms produce a successive hierarchy of clusters as a by-product. In the hierarchy, a child cluster is totally contained in its parent. Such a clustering can be represented by a *dendrogram*.

In the following sections, we will formalize the notion of cluster homogeneity. It uses the concept of *inter-observation dissimilarity measure* $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ or a *proximity measure* and the definition of the *clustering optimization criterion* [66]. The criterion will assign *cost* $f : (S_1, \ldots, S_k) \to \mathbb{R}$ for each partitioning of the training set.

2.1 Proximity measures

Intuitively, if a group of observations are close to each other and far away from the other observations, then they should form a cluster. Now, we need a formal definition of "closeness". In the case of clustering, a *dissimilarity measure* $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is a *metric* [29] if it fulfills the following axioms:

1. $d(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0$ (*non-negativity*),
2. $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = 0$, if and only if $\boldsymbol{x}_i = \boldsymbol{x}_j$ (*identity of indiscernible*),
3. $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = d(\boldsymbol{x}_j, \boldsymbol{x}_i)$ (*symmetry*) and
4. $d(\boldsymbol{x}_i, \boldsymbol{x}_k) \leq d(\boldsymbol{x}_i, \boldsymbol{x}_j) + d(\boldsymbol{x}_j, \boldsymbol{x}_k)$ (*triangle inequality*).

In clustering, triangle inequality is usually required, such a dissimilarity measures is called a *semimetric* [164]. In the rest of the thesis, we will refer to any dissimilarity function as a *distance function* and simply a *distance*.

All pairwise distances of the data set can be given a graph theoretic interpretation. Distances can be seen to form a distance matrix A, where elements a_{ij} are distances between data objects $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. It follows immediately from axiom 3 of metric spaces that A is a symmetric matrix. It can be interpreted as an *adjacency matrix* of a weighted complete undirected graph $G = (V, E)$. Here, V is the set of input vectors and E is the set of all $\binom{N}{2}$ edges. In this way, we can interpret the clustering problem as a *graph clustering* problem [157]. However, by analyzing the distance matrix, we can see that typically some vector pairs can never be in the same cluster. For that reason, as a preprocessing step, large distances are sometimes set to infinity and thus eliminated them from further processing [65].

Above mentioned simple thresholding scheme is one possibility, a more complicated scheme will compute a new *subgraph* from the original G. Subgraphs used in the literature are for example *minimum spanning tree* (MST) [30] and *k-nearest neighbour graph* (kNNG) [46]. A k-nearest neighbour graph is an undirected graph, where an edge is retained between vertices a and b if either b is one of the k-nearest neighbours of a or vise versa. An example of 1NNG in Euclidean plane is shown in Fig. 2.1. From a complete graph, a kNNG can be trivially computed in $O(N^2)$ time. In Euclidean space, it is possible to compute kNNG in $O(N \log N)$ time, but

with constants that are exponential with respect to the dimensionality of the data set [20, 28, 174]. It is an open problem whether a fast variant without large hidden constants is possible.

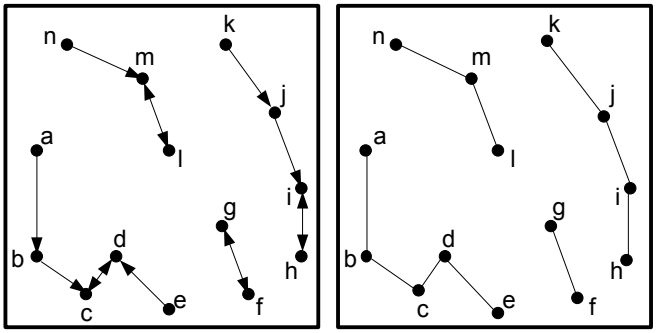


Figure 2.1: Directed 1NNG (left) and the undirected variant (right).

A commonly used metric between two vectors in a D-dimensional vector space is the l_p-norm or *Minkowski metric* [40]:

$$l_p(\boldsymbol{x}, \boldsymbol{y}) = \left(\sum_{i=1}^{D} |x_i - y_i|^p \right)^{1/p} = \|\boldsymbol{x} - \boldsymbol{y}\|_p. \tag{2.1}$$

The most common special cases are l_1-, l_2- and l_∞-norms. The l_2-norm is also known as the *Euclidean distance*. From the Minkowski metric it is natural to define the proximity measure for the clustering problem, which we call *squared error*:

$$d(\boldsymbol{x}, \boldsymbol{y}) := l_p(\boldsymbol{x}, \boldsymbol{y})^2 = \|\boldsymbol{x} - \boldsymbol{y}\|_p^2. \tag{2.2}$$

When p is set to two, we get the *squared Euclidean distance* or just squared error. Generalized variant of the squared error is used in robust statistics, where exponent 2 is replaced by a user selectable parameter α [192].

2.2 Squared error criterion

In a *centroid model*, each cluster $S_i\,((i = 1, 2, \ldots, k))$ is represented by a vector (called *centroid* or *prototype*) $\boldsymbol{c}_i \in \mathbb{R}^D$, where $C = \{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_k\}$. The goal is to find such clustering where the error given by the following criterion is minimized:

$$\frac{1}{|X|} \sum_{\boldsymbol{x} \in X} \|\boldsymbol{x} - \boldsymbol{c}_j\|_2^\alpha, j = 1, \ldots, k. \tag{2.3}$$

Fortunately, the clustering criterion can be simplified in the case of the squared error criterion; the optimal prototype $\boldsymbol{c}$ for a given vector $\boldsymbol{x}_i$ is its nearest prototype. Criterion (2.3) can be written in a form which is commonly called the *mean squared error* (MSE) [60]:

$$\frac{1}{|X|}\sum_{\boldsymbol{x}\in X}\min_{\boldsymbol{c}\in C}\|\boldsymbol{x}-\boldsymbol{c}\|^2. \tag{2.4}$$

Given a particular partitioning, the mean vector is the optimal representative to all vectors in the same cluster. If on the other hand, we set α to one, then the optimal cluster prototype $\boldsymbol{c}$ is the *spatial median* element [192]. It is found to be more robust to outliers than the mean vector but its computation is more involved [192]. It can be computed as a solution to *transportation problem*. In the rest of the thesis, we fix p and α to 2.

On the other hand, the total squared error of a cluster can be represented by [161]:

$$\sum_{\boldsymbol{x}\in S_j}\|\boldsymbol{x}-\boldsymbol{c}\|^2=\frac{1}{|S_j|}\sum_{\boldsymbol{x}_i,\boldsymbol{x}_h\in S_j}\|\boldsymbol{x}_i-\boldsymbol{x}_h\|^2, \tag{2.5}$$

where $|S_i|$ is the cardinality of the cluster S_i. In this way, the squared error clustering criterion can also be formulated as a graph clustering problem. However, in graph partitioning formulation, a cluster is represented by partitioning the labels of the observations. The squared error criterion gives another possibility to represent the solution. In particularly, it can be represented by the centroid set C. Conveniently, in the model parameter estimation setting, we can form a long *parameter vector* $\Theta=(\boldsymbol{c}_1^t,\boldsymbol{c}_2^t,\ldots,\boldsymbol{c}_k^t)^t$, as was defined in [**P4**]. A minor problem in the parameter vector interpretation is that each parameter has a fixed position in the Θ while the clustering remains the same even if their positions are shuffled [12].

2.2.1 Computational complexity

Globally and locally optimal solutions of the clustering problem posed in (2.3) are characterized by the *centroidal Voronoi diagram* (CVD) [39]. A *Voronoi tessellation* or diagram is a partition of the feature space into regions (*cells*), where each region is represented by one site (vector) called *generator*. The feature space is partitioned in such a way that each vector is mapped to its nearest generator. This mapping produces Voronoi cells that are polyhedra where each hyperplane is equidistant from two generators and the points of intersections are equidistant of at least three generators [8]. A centroidal Voronoi diagram is a structure where generators and region centroids coincide in each region. Example of the centroidal Voronoi diagram is shown in Fig. 2.2 on the right and for contrast also a Voronoi diagram that is not centroidal is shown on the left.

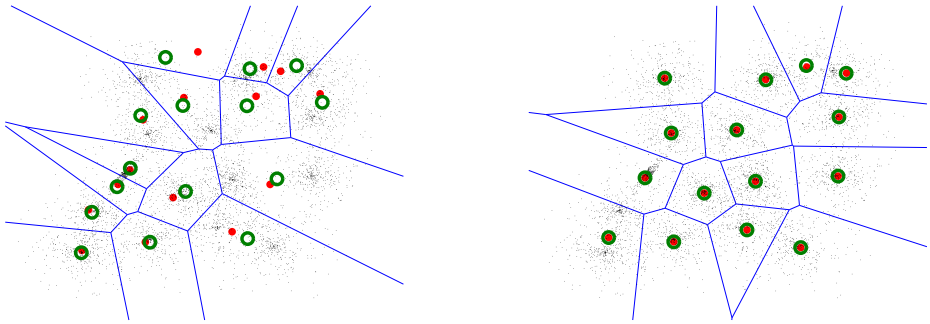


Figure 2.2: Two examples of Voronoi diagrams. A generators are marked by closed circles and region centroids by open circles. Random clustering with 2.46 MSE (left) and the corresponding CVD with 1.04 MSE (left).

Each centroidal Voronoi tessellation of the data set is a locally optimal solution, and the minimum of all locally optimal solutions is the globally optimal solution. Unfortunately, globally optimal solution for the clustering problem is not unique. For a given data set and number of clusters many different solutions might exist with the same minimum criterion value [60]. However, when the data set contains a non-overlapping clustering structure, optimal clustering is unique and one can calculate the bound on how close a given clustering is from the optimal one [131].

The number of all possible Voronoi diagrams with k generators can be shown to be $O(N^{kD})$ [82]. This result yields a polynomial time algorithm to a clustering problem when k is fixed. Enumerating all possible Voronoi diagrams turns out take $O(N^{(kD+1)})$ time [82]. On the other hand, when k is a part of the input, for example $k = f(N)$, clustering has been proved to be an NP-complete problem [38, 121]. However, when clusters are restricted to be of equal size, the problem becomes polynomial [35, 81]. Also when clustering in a 2-dimensional space, k being part of the input, the problem is NP-complete but if k is fixed, clustering is solvable in $O(N^{6k})$ time [24]. It is clear, in which ever way we view the problem, that it is not practical to find the exact solution, for that reason we have to resort to approximate or heuristic solutions of the clustering problem.

2.3 Other criteria

Many clustering criteria were summarized in [34], of which most of them were found to be NP-complete. Some clustering cost functions are briefly mentioned in the following.

Complete linkage agglomerative clustering algorithm uses as its cost function the maximum within cluster dissimilarity, which is called the *minimum diameter* clustering [65]. The objective can be formalized by finding the partitioning $P_k^* = \{S_1, S_2, \ldots, S_k\}$ from the collection of all k-partionings $\mathcal{P}_k$ such that [65]:

$$\min_{P_k \in \mathcal{P}_k} \max_{q=1,\ldots,k} \max_{\boldsymbol{x}_i, \boldsymbol{x}_j \in S_q} d(\boldsymbol{x}_i, \boldsymbol{x}_j). \tag{2.6}$$

In other words, the goal is to find such a clustering where the maximum inter-cluster distance over all clusters is minimized. It has been shown that for $k \geq 3$, the problem is NP-complete by reduction from the graph coloring problem [58, 65].

A simple algorithm to cluster graph vertices is to calculate the minimum spanning tree of the graph and cut $k-1$ longest edges from it [5, 62, 186]. MST clustering by cutting turns out to be the same as *single linkage* agglomerative clustering algorithm [62]. Edges in the MST that will be retained after clustering are called *consistent edges*, and the edges that will be cut out are called *inconsistent edges*. If the number of clusters is unknown, then one needs to find out the position in the sorted edge list where inconsistent edges end and the consistent edges start. Modeling consistent edge lengths by a truncated normal distribution, and then automatically obtaining the cut threshold was proposed in [77].

In the k-median problem [78], the goal is to find k input vectors that minimize the following expression:

$$\sum_{\boldsymbol{x} \in X} \min_{\boldsymbol{c} \in C} \|\boldsymbol{x} - \boldsymbol{c}\|^2, \tag{2.7}$$

where $C \subset X$. The problem is also known as the *discrete median* problem, in contrast to the spatial median case where prototypes are not restricted to be from the input data set. The problem is NP-complete in Euclidean space [139]. In practice, this version of clustering cost is used when the optimal prototype is either difficult or meaningless to compute. A practical heuristic in that case is the *partition around medoids* (PAM) method [93], which is also known as the k-medoids algorithm. It is otherwise exactly the same as k-means, but the step for determining the optimal centroid has been replaced by a step where such an input element in the cluster is selected as the prototype that it minimizes the cluster distortion (2.7).

A clustering cost function that is specifically designed for hierarchical clustering is proposed in [45]. The goal is to generate such a graph partitioning where the length of the minimum spanning tree inside each cluster is minimized. It is shown in [45] that such a clustering problem is NP-complete. An approximation algorithm is also proposed that working in $O(N \log N)$ time and is within a factor of 3.42 from optimal clustering.

Previously, we have considered a case where the weights of the graph consist of real numbers, as defined by the l_p distances. However, one can also cluster

unweighted graphs, for example when clustering a social network data. In that case, distances between the observations are binary, a person $\boldsymbol{x}$ either knows a person $\boldsymbol{y}$ or does not. The squared error criterion is not sound in this case. Possible clustering criteria for this kind of data are *modularity cost* [15] and a objective function based on *dominant sets* [141]. The graph clustering problem is NP-complete for both of these objectives.

2.4 k-means algorithm

Perhaps, the most well-known clustering algorithm is the *k-means* [123], mainly because of its simplicity. It is also known as *generalized Lloyd algorithm* (GLA) [115], *hard c-means* and *Linde, Buzo, Gray* (LBG) algorithm. It has been applied in numerous different fields, such as statistics [165]. It is a practical solution to the optimization problem posed in (2.3).

The operation of the k-means algorithm is based on the fact that a globally optimal solution has to fulfill two *necessary* conditions [60]:

- *Nearest neighbour condition*: given cluster centroids C, optimal assignment of a data vector to the cluster is to its closest cluster prototype:

$$\boldsymbol{c}^* = \arg\min_{\boldsymbol{c}\in C} \|\boldsymbol{x} - \boldsymbol{c}_j\|. \tag{2.8}$$

- *Centroid condition*: given a partition $S_i = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{|S_i|}\}$, the optimal prototype is the partition centroid:

$$\boldsymbol{c}_i = \frac{1}{|S_i|} \sum_{j=1}^{|S_i|} \boldsymbol{x}_j. \tag{2.9}$$

There are several different strategies to prove the centroid condition (see [60] for three proofs). An interesting proof, which does not employ a direct use of derivatives, is obtained by considering the following equality [161]:

$$\sum_{i=1}^{|S_j|} \|\boldsymbol{x}_i - \boldsymbol{z}\|^2 = \sum_{i=1}^{|S_j|} \|\boldsymbol{x}_i - \boldsymbol{c}_j\|^2 + |S_j| \|\boldsymbol{c}_j - \boldsymbol{z}\|^2, \tag{2.10}$$

where $\boldsymbol{z}$ is any vector in $\mathbb{R}^D$. Since, the last term is non-negative, the minimum distortion is obtained when $\boldsymbol{c}_j = \boldsymbol{z}$.

The third necessary optimality condition is called *zero probability condition* [60]. It states that observations at the Voronoi face, occur at zero probability. If it

happened, there would be more than one cluster where the input vector could belong to, and the order of processing the data set will affect the solution. In practice, with real valued feature vectors the zero probability condition is true.

The k-means algorithm is a *local search* heuristic where the current solution is iteratively improved by alternating between nearest neighbour condition and centroid condition until convergence. When converged, centroids are generators of the centroidal Voronoi diagram [39]. The algorithm starts from an initial solution C, which can be randomly drawn vectors from X, or an output of any clustering algorithm minimizing the squared error. The effectiveness of different initialization strategies has been studied empirically for example in [142]. In case of more advanced clustering techniques, everything can be seen as an initialization to k-means, which as a local search heuristic can never give a worse solution. In any iteration, a lower bound on the locally optimal solution can be calculated in $O(N \log N + ND)$ time [191].

Convergence in finite number of steps follows from the fact that in each iteration the cost strictly reduces, so each solution can be seen only once during the execution of the algorithm. Since each solution induces a Voronoi diagram, for which we already have upper bound $O(N^{kD})$, k-means will converge in at least this many iterations. However, convergence to a globally optimal solution is not guaranteed. Actually, examples can be generated where the performance of the k-means is arbitrary bad, depending on the initialization [4, 67]. In practice, it is better to run k-means multiple times in parallel with different random initialization, and pick the solution with the least cost. This approach is known as the *multi-start k-means* [70]. The algorithm can be made significantly faster by using the lower bound on the locally optimal solution and stopping early if the current random start will not provably lead to a better solution than the best of previously computed solutions [191].

The time complexity of the standard version of k-means is $O(NkI)$, where I is the number of iterations needed for the convergence. It is an interesting task to try to analyze the worst case and the average case convergence speed of the algorithm. Har-Peled and Sadri [67] analyzed the one-dimensional case and found out that the upper bound on the number of iterations $O(N\Delta^2)$, does not depend on the number of clusters, but it depends on the spread of the data set Δ. In one-dimensional case, if the distance between the closest pair is normalized to unity, Δ is the diameter of the input data set. Recently, Arthur and Vassilvistkii [3] constructed an example, where k-means takes $2^{\Omega(\sqrt{N})}$ iterations.

In [14], it was proven that the convergence behaviour of the k-means is the same as Newton optimization algorithm. For a quadratic cost function, Newton algorithm finds the optimal solution in one step. Unfortunately, in finding a non-quadratic case locally optimal solution demands a super-linear time, and this analysis applies to k-means [14]. It has been noted that the convergence speed is affected by the initializa-

tion strategy, if initialization is performed more intelligently than random sampling from uniform distribution, then less iterations are needed for convergence [4].

Standard implementation of the k-means uses $O(kN)$ time per iteration. The nearest centroid search can be performed faster by placing the centroids to a spatial indexing structure [61, 90, 120, 134, 144]. Although, spatial indexing structures such as kD-tree [57] do scale well to a large number of cluster centroids, they do not scale to a high-dimensional case when $D > 8$ [144]. Then nearest centroid search will approach linear time instead of logarithmic time as promised by the search structure. The number of distance computations can also be reduced by considering that movement of a centroid is a *local* operation and it affects the neighbouring clusters only. A method utilizing this property is known as *code vector activity detection* [95]. Using the triangle inequality, bounds for both the partition and the centroid step can be computed resulting additional saving in the number distance calculations [43].

2.5 Exact and approximation algorithms

In addition to the exact clustering algorithm proposed by Inaba *et al.* [82], a number of other exact algorithms have also been proposed. One of the first exact algorithms for the squared error clustering problem was a dynamic programming algorithm by Jensen [85]. A branch-and-bound algorithm was proposed in [55, 107]. An optimization theoretic approach for (2.3) was formulated as an integer programming problem [150] and a constrained hyperbolic 0/1-program [132]. An interior point method was used in [132] to solve the problem with the help of branch-and-bound and numerous other mathematical programming tools. Interestingly, the authors managed to obtain optimal clustering to fairly large data sets, e.g. Fisher's iris data set consisting of 150 vectors. A method with much simpler implementation was designed by Brusco [18]. His method uses the repetitive branch-and-bound to achieve similar efficiency as in [132]. In [145], it was proven that the clustering problem can be formulated as a concave minimization problem, where every local minima is an integer solution. Relaxed version of the above mentioned algorithm yields a non-optimal but practical algorithm [145].

However, some recent theoretical work in approximation algorithms has shown that, at least theoretically, a *polynomial time approximation scheme* (PTAS is possible for the clustering problem [1, 49, 129, 137]. A PTAS is an algorithm, which takes as its input a data set and a user selected parameter ϵ. Parameter ϵ gives the desired maximal error of the approximation. A PTAS has polynomial time complexity, with all values of ϵ [176]. Unfortunately, none of the PTAS clustering algorithms have turned out to be practical because of large hidden constants [67] and, therefore,

practitioners use heuristic algorithms without performance guarantees. There is a big gap between theory and practice in the field of clustering algorithms [137].

On the other hand, practical clustering algorithms with proven worst case approximation ratio, have been published. *Randomized local search* (RLS) [53] is a clustering algorithm where local optima are avoided by perturbing the solution by swapping one centroid to a new location. A deterministic variant of the swapping algorithm was analyzed in [91] and it turns out to be $(25 + \epsilon)$-approximation algorithm. However, authors also propose a multi-swap variant, which achieves $(9 + \epsilon)$-approximation in the worst case [91]. Improving the randomized initialization of the k-means algorithm yields in terms of expectation an $8(\ln k + 2)$-approximation algorithm [4].

2.6 Agglomerative algorithms

An agglomerative clustering [13, 88] algorithm constructs the clusters by a sequence of merge operations. The agglomeration process starts by initializing each data vector as its own cluster. Two clusters are merged at each step and the process is repeated until the desired number of clusters has been obtained. *Ward's method* [182], also known as *pairwise nearest neighbour* (PNN) [47], selects the cluster pair to be merged in such a way that the square error criterion is least increased. In total, $O(N)$ iterations are needed, and in each iteration just enumerating all pairwise distances take $O(N^2)$, and so yields $O(N^3)$ time complexity in total. When considering the operations in the iterations, it turns out that most computation originates from distance calculations [158]. For arbitrary dimension D, this leads to $O(DN^3)$ time complexity. We call this method *exact PNN* [47].

Several speed-ups have been proposed in the literature to the exact PNN. One possibility to reduce the distance calculations is to use a matrix of pairwise distances. The matrix is upper triangular as the merge costs are symmetric. When two partitions S_i and S_j are merged, only row i and column j must to be recalculated, which leads to $O(N)$ update operations. Authors in [177] propose to use exact PNN variant with distance matrix for color quantization. Kurita [111] proposed to use distance matrix for updates and a heap structure for finding the minimum pairwise distance in $O(\log N)$ time. The time complexity for Kurita's method is $O(N^2 \log N)$, but unfortunately, keeping all pairwise distances in the heap and in the distance matrix the causes space complexity to $O(N^2)$.

Fast exact PNN [52] has linear space complexity, but is still an order of magnitude faster than the previous versions of PNN. Here, the idea is to maintain the nearest

neighbour pointer nn() to all partitions:

$$\mathrm{nn}(S_a) = \arg \min_{i \in [i,N], i \neq a} d_{\mathrm{PNN}}(S_a, S_i) \tag{2.11}$$

and use it to find the cluster pair to be merged fast. To find the minimum distance only $O(N)$ steps are needed in each iteration. After a merge operation, the nearest neighbour pointers must be updated to point to a new possibly closer cluster. This method is faster than the previous ones but its time complexity is still $\Omega(\tau N^2)$, where τ denotes the number of partitions whose nearest neighbour pointer must be updated. A similar idea was presented in [135, 42]. In their approach, a cluster pair to merge is searched from on the list of nearest neighbour pointers, and such clusters pairs (S_a, S_b) are merged which are $\mathrm{nn}(S_a) = b$ and $\mathrm{nn}(S_b) = a$. Unfortunately, the authors did not provide any time complexity analysis of their method.

Another possibility to speed up the operation of PNN is to use a dynamic closest pairs data structure, where update cost is $O(N \log^2 N)$ [44]. The total time complexity for clustering is then $O(N^2 \log^2 N)$ and the space complexity is $O(N)$. A Speed-up of around 35% can also be gained compared to the fast exact PNN by deferring the distance computations until necessary [94, 25]. Practical methods to achieve 10 to 15% reductions in running time have been proposed in [180]. All these methods still require quadratic time.

The $O(N^2)$ time complexity barrier can be broken by introducing an approximation to the search for the minimum merge cost [**P1**]. The idea is to restrict the search for the closest cluster pair on the k-nearest neighbours of each cluster. The time complexity of the algorithm is then improved from $O(\tau N^2)$ to $O(\tau N \log N)$ at the cost of a slight increase in distortion.

2.7 Graph clustering

There also exists a number of graph clustering algorithms [157] that are based on graph theoretic properties instead of optimizing some defined optimization criterion. Properties that have been used are for example, *connectivity* [17], *Szemerédi Regularity* [162] and *scale-free minimum spanning tree* [138].

Graph clustering based on connectivity starts with taking the complete graph and restricting it based on some rule. A typical restriction is the k-nearest neighbour graph. The clusters are then defined to be the connected components of the graph [17]. *Mutual k-nearest neighbour* graph (MkNNG) based clustering [17] defines a new graph based on a directed k-nearest neighbour graph where directed edges are turned to undirected by removing all those edges that point to one direction, only. Then, connected components of the graph are clusters and isolated vertices are outlier points. It has been shown that if clusters are separated, setting $k = \log(N)$

will give a correct clustering [17]. MkNNG clustering algorithm was generalized to a mutual range graph in [27]. *Range graph* is a proximity graph where all data objects that are closer than a user specified threshold from the object in question are considered neighbours. In this variant of MkNNG, a directed range graph is calculated first and it is then converted into a mutual one.

In CHAMELEON [92], a k-nearest neighbour graph is formed from the similarity matrix. Clustering is then performed in two stages: the k-nearest neighbour graph is divided into a relatively large number of sub-clusters and small sub-clusters are then iteratively processed with agglomerative clustering to obtain the final clustering. Graph theoretic clustering algorithm by Jarvis and Patrick [84] on the other hand defines, instead of kNNG, a *shared neighbourhood* graph. Shared neighbourhood between two elements stands for the number of same elements contained in their k-nearest neighbour lists. As in CHAMELEON, clustering is then performed on the shared neighbourhood graph. In a sense, the shared neighbourhood graph is a weighted version of MkNNG.

2.8 Gaussian mixture modeling

Another way to think of clustering is to assume that the data set has been sampled from a parametric generative distribution. This is called *model clustering* or in the statistics literature *density estimation*. The most common mixture model is *Gaussian mixture models* (GMM) [12], which is our concentration from here onwards. The most popular cost function is *maximum likelihood* (ML), where such a model is found that maximizes the likelihood of the observations. Gaussian mixtures can also be estimated by optimizing some information theoretic criteria, such as *minimum message length* (MML) [51]. By minimizing the message length, it is possible to simultaneously solve the number of components and the GMM parameters. In the following, we will only discuss the maximum likelihood case.

In this section, we assume that data has been sampled from k multivariate Gaussians, which together form a mixture model. Each Gaussian component is parametrized by its prior weight, mean vector and covariance matrix $\theta_i = (\pi_i, \boldsymbol{\mu}_i, \Sigma_i)$, and the full parametrization of the whole model is $\Theta = (\theta_i)_{i=1}^{k}$. Diagonal covariance matrices are mostly used in situations where there is a need to optimize the speed of the computation. The reason is that, computing the log-likelihood needs the inverse of the covariance matrix.

We also make the so called *incomplete data* assumption, where each data vector is assumed to be generated by one of the Gaussians but the class label is not available, hence the term incomplete data. The form of the Gaussian probability density

function is:

$$p(\boldsymbol{x}|\Theta) = \sum_{i=1}^{k} \pi_i p(\boldsymbol{x}|\theta_i), \tag{2.12}$$

where $\boldsymbol{x}$ is a vector in $\mathbb{R}^D$. Then, given N independent and identically distributed (i.i.d.) samples, the complete-data log-likelihood is expressed as:

$$\begin{aligned} \log p(X|\Theta) &= \log \prod_{j=1}^{N} p(\boldsymbol{x}_j|\Theta) \\ &= \sum_{j=1}^{N} \log \sum_{i=1}^{k} \pi_i p(\boldsymbol{x}_j|\theta_i), \end{aligned} \tag{2.13}$$

where $\sum_{i=1}^{k} \pi_i = 1$. Now, the maximum likelihood parameter estimate $\hat{\Theta}_{\mathrm{ML}}$ can be expressed as:

$$\hat{\Theta}_{\mathrm{ML}} = \arg\max_{\Theta} \log p(X|\Theta) \tag{2.14}$$

Unfortunately, no closed form solution exists to (2.14), and therefore an algorithmic solution has to be devised to solve the problem. A locally optimal iterative scheme is known as *expectation maximization* (EM) algorithm [37]. Even though global optimality of the ML cost function cannot guaranteed [12], something is known theoretically about the global optimality. If Gaussian components are well-separated, then the mean vectors can be tractably estimated by projection based methods [2, 31, 178] and even using EM [32].

EM can also be seen as a generalization of k-means algorithm [12]. It works by alternating between two steps, finding the posterior probabilities (*responsibilities*) for each observation and re-estimating the model parameters given previously obtained responsibilities [12]:

- **E step:** evaluates responsibilities using the current parameter values:

$$\gamma(z_{ij}) = \frac{\pi_j N(\boldsymbol{x}_i|\boldsymbol{\mu}_j, \Sigma_j)}{\sum_{j=1}^{k} \pi_j N(\boldsymbol{x}_i|\boldsymbol{\mu}_j, \Sigma_j)}. \tag{2.15}$$

- **M step:** Re-estimates the model parameters given the current responsibilities:

$$\boldsymbol{\mu}_j^{\mathrm{new}} = \frac{1}{N_j} \sum_{i=1}^{N} \gamma(z_{ij}) \boldsymbol{x}_i, \tag{2.16}$$

$$\Sigma_j^{\mathrm{new}} = \frac{1}{N_j} \sum_{i=1}^{N} \gamma(z_{ij}) (\boldsymbol{x}_i - \boldsymbol{\mu}_j^{\mathrm{new}})(\boldsymbol{x}_i - \boldsymbol{\mu}_j^{\mathrm{new}})^T, \tag{2.17}$$

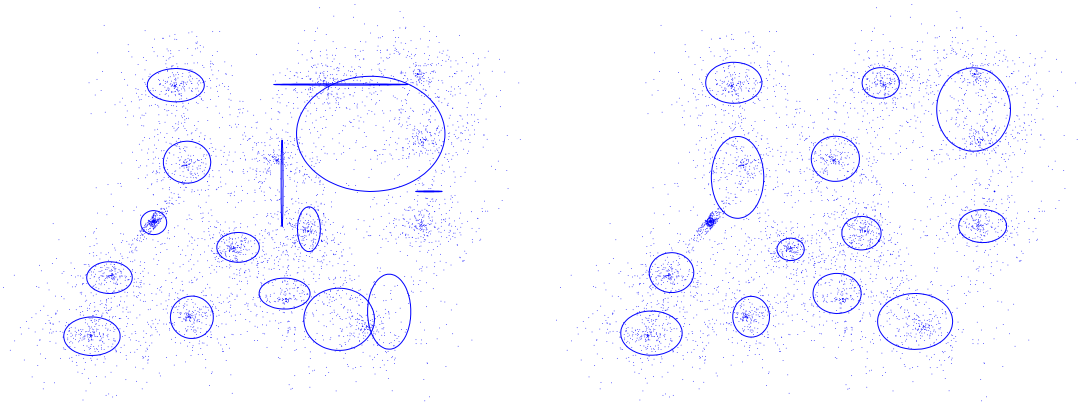
$$\pi_j^{\text{new}} = \frac{N_j}{N}, \tag{2.18}$$

where

$$N_j = \sum_{i=1}^{N} \gamma(z_{ij}). \tag{2.19}$$

In the above, $\gamma(z_{ij})$ is the responsibility of vector $\boldsymbol{x}_i$ belonging to the Gaussian component j. It can be seen as the generalization of the discrete partition labeling of the hard clustering. In the hard clustering case, $\gamma(z_{ij}) = \{1, 0\}$; it is one when the observation belongs to the cluster, and zero when it does not.

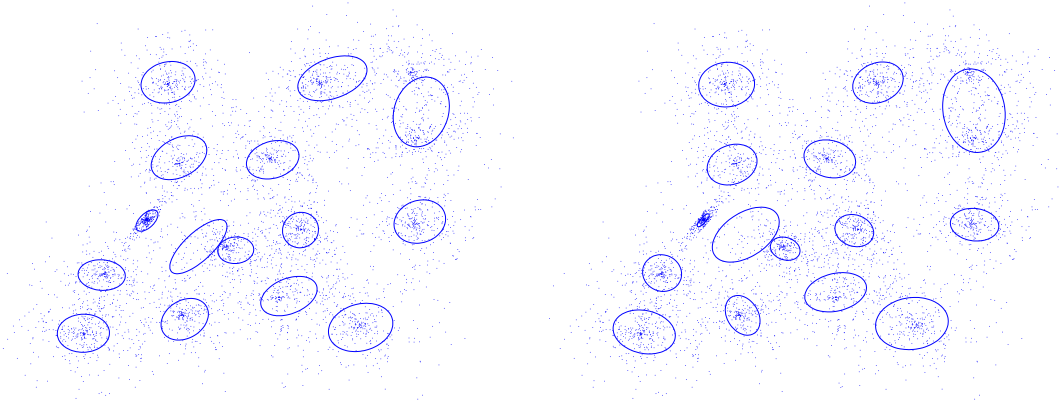
The EM algorithm provides a locally optimal solution to the log-likelihood maximization problem. The quality of the solution generated by EM depends on the initial solution [130]. To overcome this problem, a number of heuristics have been proposed. The most common way to initialize the EM algorithm is to first run hard clustering algorithm (e.g. k-means) on the data set and then to interprete each cluster as a Gaussian component. Component parameters are calculated from relative numbers of observations (mixing weight) and centroid (mean vector), and the covariance matrix is calculated from the observations in the cluster.



$\log p(X|\Theta) = -26.8$ $\log p(X|\Theta) = -26.6$

Figure 2.3: Example of GMM with diagonal covariance matrices: random initialization tuned by 10 k-means iterations (left) and a locally optimal output of the EM algorithm (right).

The *Split-and-merge EM* (SMEM) algorithm [173] performs the EM algorithm after local convergence and selects three components for a split-and-merge procedure. One of these component is split in two halves, and the other two components



$\log p(X|\Theta) = -26.6$ $\log p(X|\Theta) = -26.5$

Figure 2.4: Example of GMM with full covariance matrices: random initialization tuned by 10 k-means iterations (left) and a locally optimal output of the EM algorithm (right).

are merged. Minagawa *et al.* [133] noted that the acceptance rule of the SMEM algorithm is incorrect, resulting to an accidentally discarded globally optimal solution. Merge is defined in a theoretically sound manner in the SMEM algorithm, whereas split is a heuristic operation. For this reason, a merge based algorithm was proposed by Verbeek *et al.* [179]. Their algorithm iteratively inserted components into the mixture until a desired number of Gaussian components is obtained. After each merge, EM is run again until convergence.

Genetic algorithms (GA) have also been used to estimate the GMM parameters [128] when maximum likelihood is used to measure the fitness of a solution. A genetic algorithm using *Minimum description length* (MDL) based fitness function has also been proposed in [146] where the number of components is automatically set. In these approaches, GMM parameters were encoded into the chromosome, but a different encoding is proposed in [166]. A chromosome codes the indices of the generating mixture components for the vectors. In a sense, the proposed approach tries to estimate the Gaussian mixture parameters directly by encoding the latent variables in the individuals of the GA.

It has been noted that sometimes EM converges slowly to a locally optimal solution [50, 26]. Especially, EM will slow down considerably if it hits almost the plateau part of the fitness landscape [163]. Alternative strategies that maximize the parameters sequentially, instead of simultaneously, have been proposed as *Space-alternating generalized EM* (SAGE) [50] and *Component-wise EM* [26].

2.9 Outlier detection

Outlier detection refers to methodologies for eliminating observations from the input data set that adversely affect the statistical modeling [76]. *Inliers*, on the other hand, are the observations one is looking for, and those should be retained for the modeling. Numerous clustering methods include outlier detection as an additional step in the modeling process [17, 41, 63, 64, 76, 86, 189]. In general, an input data object is designated as an outlier if it does not *fit* to the model being constructed [76].

Another point of view to outlier detection is to see them not as noise to be eliminated but as interesting and surprising observations in need of more study. This approach is taken in data mining and knowledge discovery. A mixed case can also happen, where inlier observations are corrupted by noisy observations, and the task is still to find surprising observations [119]. Some applications of outlier detection in knowledge discovery include intrusion detection in computer network security [112, 114, 140] and abnormal human-activity detection [188].

Outlier detection methods can be divided into two categories: *supervised* or *unsupervised*. Supervised methods require external training data, where data vectors are labeled as inliers or outliers. Supervised outlier detection is also known as *novelty detection* [127]. Examples of the two-class classification approach applied to outlier detection are for example *neural networks* [118, 184], neural networks with *radial basis function* (RBF) with *principal component analysis* (PCA) *dimensionality reduction kernel* [117] and *k-NN classifier* [114].

The two-class classification approach has problems; it is implicitly defined that all relevant types of outliers have already presented in the training data set. One-class training is the other possibility where only examples from the inlier class are used in the training. In the classification step, an unknown observation is scored against the inlier class and the observation is decided to be an outlier if its score is below a user specified threshold. These methods are for example one-class *support vector machines* (SVMs) [122, 167] or one-class SVM with kernel *maximum likelihood linear regression* (MLLR) adaptation [188]. A hybrid method is one where the classifier does not rely on outlier examples on training step but can improve the classification accuracy if examples are available [23].

In unsupervised outlier detection, we have a data set without any separate training set with labeled examples. In practice, unsupervised methods need an exact definition of what is an outlier. The differences between the methods are then mostly related to the differences in the definitions. Intuitively, an outlier should be an observation that is far from other observations, as an example see Fig. 2.5. The figure represents astronomical observations of different stars. Ground truth labeling of the observations to inliers and outliers was done by a trained astronomer. Inlier observations represent stars in the so called main sequence and outliers are stars,

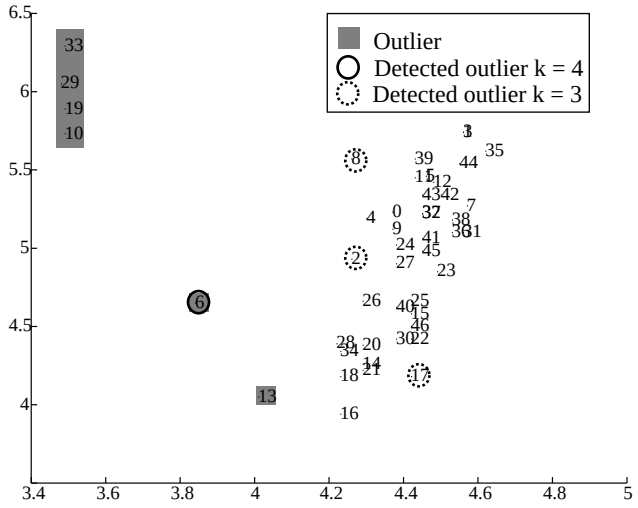


Figure 2.5: Example of a data set with normal observations and an outlier observations. Shown also outliers detected by [**P2**] with two different neighbourhood sizes.

which do not fit to that pattern.

Definitions of unsupervised outliers fall roughly into five categories [87]: i) *distribution-based*, ii) *clustering-based*, iii) *depth-based*, iv) *distance-based* and v) *density-based*. Distribution-based methods originate from statistics where an observation is considered as an outlier if it deviates too much from the underlying distribution. For example, in normal distribution an outlier is an observation whose distance from the average observation is three times of the variance [56]. The problem is that in real world cases the underlying distribution is usually unknown and cannot be estimated from the data without outliers affecting the estimate. Clustering-based methods work similarly: the whole data set is clustered and observations that do not fit to the overall clustering pattern are decided as outliers [86, 187, 190][**P3**]. As in distribution-based methods, outliers affect the clustering model. Depth-based outlier detection methods compute different layers of D-dimensional convex hulls of the data set. Outliers are then defined to be in the outer layers of these hulls [87].

Distance-based methods [105, 106] define an outlier as an observation that is at least at distance $d_{\min}$ away from p percentage of observations in the data set. The problem is then finding an appropriate settings of $d_{\min}$ and p such that outliers would be correctly detected with a small number of false detections. This process usually needs domain knowledge [106]. Typical distance-based methods require

$O(N^2)$ distance computations, but a faster implementation has been reported in [9].

In density-based methods, outliers are detected from local density of observations. These methods use different density estimation strategies. A low local density of the observation is an indication of a possible outlier. Ramaswamy *et al.* [147] proposed a method, in which n largest kNN distances are considered as outliers. This can be seen as *sparseness estimate* of a vector, in which the n sparsest vectors are considered as outliers. Other methods based on k-neighbourhood are presented in [16, 9, 87, 108, 190]. Graph theoretic methods infer the local density by using a k-nearest neighbour graph [17][**P2**]. As with distance-based methods, computation of the local density for all observations can be slow. Speed-up can be obtained by pruning the nearest neighbour search [87].

Chapter 3

Speaker Recognition

IN *speaker verification* [11], an unknown speech utterance is introduced to the system accompanied by a claim. The task is to decide whether the claim was true or false, by matching the unknown test utterance to a previously stored model. In *speaker identification*, on the other hand, the unknown speech utterance is matched against a database of known speakers. Output of the speaker identification system is the identity of the speaker in question, or in the case of *open-set identification*, the decision that the speaker is unknown to the system. Speaker recognition tasks can be further classified into *text-dependent* and *text-independent* tasks.

In text-dependent systems, the spoken text or password is fixed. In text-independent recognition, the user is free to speak anything as the enrollment utterance, and also in the using of the system. Applications of the text-dependent recognition are restricted to security and access control whereas text-independent recognition can be used in continuous recognition as well.

A speaker recognition system consists of two main components: i) *front-end* processing component and ii) pattern recognition component. Front-end processing reads audio data, preprocesses it, applies voice activity detection and feature extraction. The pattern recognition subsystem, handles features modeling and pattern matching. Feature extraction produces a sequence of feature vectors $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\} \subset \mathbb{R}^D$. Each feature vector describes short-term signal statistics. The feature cloud from the utterance contains speaker discriminative information. The front-end system and pattern recognition system can be separately tuned.

3.1 Front-end processing

3.1.1 Voice activity detection

In the general terms, *voice activity detector* (VAD) [10] is a binary classifier that labels segments of the signal either as speech or non-speech. It is usually needed as the first component in voice-based applications such as automatic speech recognition, speech enhancement [148], forensic speech skimming [172] and speaker recognition [152]. A common property of these applications is that only human sounds (typically only speech) are of interest to the system, and it should therefore be separated from the background. In different applications, the granularity of the voice activity decision varies. We call *frame level VAD* a system which produces per frame decisions, and *end-point detector* a system which detects continuous speech segments.

Voice activity detection is an essential part of any speaker recognition system [169]. It is clear that including non-speech frames in the modeling process would bias the resulting model, especially if the number of non-speech frames is significant. It is also known that not all speech frames have equal discriminative power as voiced phonemes are more discriminative than unvoiced phonemes [156]. It has been found that depending on the speech corpus, a badly optimized VAD can lead to a catastrophic detection accuracy: from 17% error rate to nearly the accuracy of just coin flipping [**P7**]. A similar result has been recently seen in the *speaker diarization systems* [171] where the biggest source of diarization error is contributed by errors made by the VAD preprocessing component [80].

Numerous approaches for voice activity detection exist in the literature and exhaustive categorization of all those approaches is not easy. However, voice activity detectors fall into two categories: i) those that threshold short-term signal statistics, and ii) those that use machine learning approaches. The VADs in the first group perform thresholding according to *spectral likelihood ratio* [89], *periodicity* [**P7**], *energy* [170] and *long-term spectral divergence* (LTSD) [148] just to mention a few.

3.1.2 Feature Extraction

In speaker recognition, features that capture the anatomical and behavioral characteristics of the speaker are desired. Speech itself is highly complex signal, which carries several features mixed together [154]. Because of the high complexity, it is difficult to realize the features that robustly capture the speaker individuality [97].

Features can be divided into three categories: *segmental*, *suprasegmental* and *high-level* [151]. Segmental features contain speech signal characteristics computed over short segments of 10-30 milliseconds in duration. Suprasegmental features, on

the other hand, contain information that spans multiple segments, sometimes the whole utterance. High-level features are discrete in nature, for example linguistic features such as idiosyncratic word usage or phonetic features such as pronunciation style. In the following, we concentrate on the segmental and suprasegmental features.

3.1.3 Segmental features

In segmental feature extraction, the speech signal is analyzed in short segments or *frames* where local stationarity is assumed. Frame length is typically 10-30 milliseconds, and frames overlap 25-50% of the frame length [98]. A single feature vector is computed from each frame individually. The *mel-frequency cepstral coefficients* (MFCCs) [33] are the most popular spectral features used in speaker recognition systems [19].

The steps in the MFCC processing are summarized in Fig. 3.1. Time-windowing is performed first to suppress discontinuities at the frame boundaries, followed by the magnitude spectrum computation using *discrete Fourier transform* (DFT) [136]. Magnitude spectrum is obtained from the complex valued Fourier coefficients by taking the absolute value of it. Filterbank processing provides dimensionality reduction by creating a smoother version of the original magnitude spectrum. Non-linear frequency warping is obtained by placing the center frequencies of each filter according to the defined warping function. It is designed so that it will emphasize lower frequency part of the spectrum. Warping function used in MFCC is the so called *mel* or *melody* scale [69]. The cepstrum is finally obtained by first applying logarithm and then the *discrete cosine transform* (DCT). The zeroth coefficient corresponds to the frame energy, and it is dropped. Typically 10-20 low-order coefficients are retained for further processing.

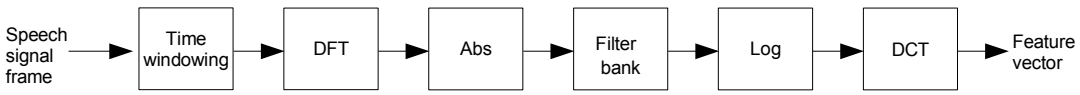


Figure 3.1: Work flow of the MFCC signal processing.

3.1.4 Suprasegmental features

Suprasegmental features are typically calculated over the whole utterance. *Long-term average spectrum* (LTAS) [79] is defined as an average magnitude spectrum of the whole utterance. An estimate of the average magnitude spectrum can be computed by first dividing the speech signal into overlapping frames and then computing the magnitude spectrum for each frame separately, followed by time averaging [183].

LTAS is a n-dimensional feature vector where n is the FFT binsize. It has been mostly used in forensic speaker recognition [155] as it is easy to compare two different LTAS vectors [116] visually. Example on how different LTAS vectors can be visually evaluated is shown in Fig. 3.2. It shows four LTAS vectors calculated from four different subjects over telephone line. It has also been used in early automatic speaker recognition studies [126, 68]. However, LTAS is of lower accuracy than MFCC, and does not provide substantial improvement when used in combination with (classifier fusion) MFCC [**P6**]. In terms of speaker modeling and matching, LTAS is straightforward to compute. LTAS vectors can be directly matched using any similarity or dissimilarity measure. Similarity functions that have been used in LTAS based speaker recognition include Euclidean distance, cosine similarity, correlation and Kullback-Leibler divergence [**P6**] and also a recent study [175] used *standard deviation of difference distribution* (SDDD) [68] index.

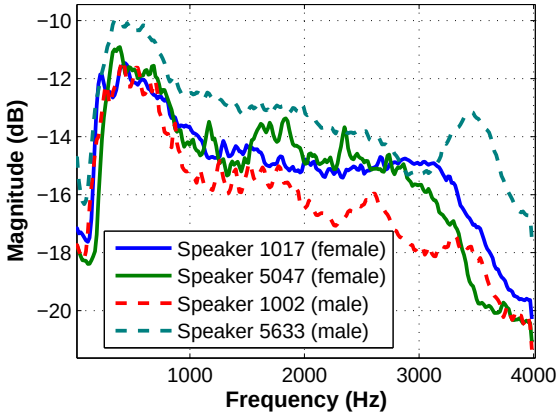


Figure 3.2: Example of LTAS calculated from four different speakers.

Long-term features calculated from the so called *prosodic* parameters have also been used in speaker recognition. Prosody refers to the rhythm, stress and intonation of speech. Most common prosodic features describe syllable length, loudness and pitch (or F0). F0 and intensity contours have been used in text-dependent speaker recognition [6]. In the text-independent case, long-term F0 distributions contain speaker discriminative power [99]. Distributions can be matched, either by their parameter vectors or by histogram matching [99]. Histogram matching with Kullback-Leibler divergence was found to be more accurate. However, a histogram is a discrete model of the long-term distribution and a continuous model such as kernel density estimator could provide better accuracy [104]. In [36], prosody features were extracted by first segmenting the whole utterance into pseudo-syllables using energy contour. From each pseudo syllable, a sixth-order Legendre polynomial was fitted

to the log F0 and log energy contours. Legendre polynomial coefficients, in addition to segment duration, were then modeled by GMM using a session variability compensation [96].

3.2 Speaker modeling

Speaker recognition systems have typically used generative models such as vector quantization [11, 160] (aka the *centroid model*) and Gaussian mixture models [153], or discriminative models such as *support vector machines* (SVMs) [21] and *neural networks* [48]. Approaches that combine the generative and discriminative models have been recently introduced [22]. When using generative models we assume that there is an unknown distribution where the feature cloud is sampled from. The goal in the modeling is then to estimate the parameters as well as possible. In discriminative training, the goal is to find *decision boundary* that classifies unseen samples as well as possible. The basic difference is that discriminative modeling needs examples from the negative (or *impostor*) class, whereas generative approach works only with the speakers own feature vectors. In the following, we restrict the discussion to the generative models.

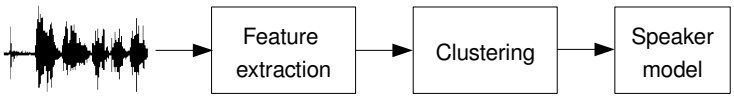


Figure 3.3: Speaker model training in the maximum likelihood approach.

A generative model is typically trained using the *maximum likelihood* (ML) principle where the speaker model is optimized by only analyzing the training data provided by the user. In the case of GMM training, statistical independence of the feature vectors is assumed, and the log-likelihood (2.14) is therefore used. In VQ, mean squared error (2.4) is the cost function to be optimized. System diagram of the maximum likelihood approach is shown in Fig. 3.3. The ML approach usually does not generalize well to unseen speech data with a finite amount of training material [12].

The size of the centroid set C (number of Gaussian components, in the case of GMM) defines the *model order*, and it is a user-selectable parameter. Increasing the model order creates more accurate estimation of the underlying distribution and decreases recognition errors with increased processing time. Too large models will over-fit the training data and degrade the speaker recognition accuracy [102, 103].

Maximum a posteriori (MAP) training [59] attacks the problem of limited training data by restricting how closely the model is allowed to be optimized for the

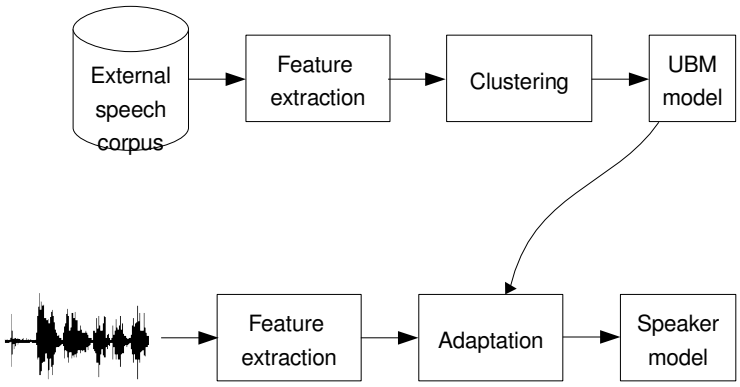


Figure 3.4: Speaker model training by the MAP principle.

training data. In the MAP approach, prior knowledge of the distribution of the model parameters is incorporated into the modeling process. Even if some areas of the feature space are less represented in the training data, the prior information about the parameters can help to overcome the problem. However, incorporating the prior information is not trivial because prior parameter distribution has its own parameters, known as *hyperparameters*, which can be difficult to estimate. In the GMM case, *universal background model* (UBM) [152] was proposed as a practical way to set the hyperparameters. In VQ, similar approach with UBM as a hyperparameters to the centroid set is proposed in [**P4**]. A system diagram of the maximum *a posteriori* approach using UBM-based prior distribution parameters is shown in Fig. 3.4.

The UBM for GMMs is calculated by optimizing the maximum likelihood principle using EM algorithm. In VQ, the UBM is calculated by optimizing the MSE cost function. Individual speaker models are then adapted from the UBM using a modified MSE criterion by applying k-means or EM algorithm.

3.3 Mismatch compensation

Errors in speaker recognition accuracy are mostly due to mismatch between testing and training conditions. With TIMIT corpus, which can be obtained from *Linguistic Data Consortium* (LDC)[1] , which contains only noise free and single microphone recordings, a MFCC based system is able to achieve 0% error rate [97]. On the other hand, when channel mismatch is present, the error rate can increase considerably. For a practical speaker recognition system it is necessary to include some form of mismatch compensation technology.

[1]http://www.ldc.upenn.edu/

Various approaches have been proposed for tackling the channel mismatch problem, including robust feature extraction [125], feature normalization [143], model transformation [96, 181, 168], and match score normalization [7, 152]. Feature and speaker model transformations, including *affine transformation*, have been studied by different authors [124, 125, 181, 159]. Most common mismatch compensation techniques are *eigenchannel* [96] for Gaussian mixture models and *nuisance attribute projection* (NAP) [19] for SVMs. In both cases the model parameters are tuned using the parameters of the channel and session variability transformation.

Above mentioned channel compensation methods usually require either parallel training data recorded simultaneously through various handsets, or a large number of training utterances collected from multiple recording sessions from a number of speakers. These data sets are then used for estimating the transformation parameters. Instead of learning the transformation parameters from an external corpus, another approach is to design a matching scheme that is invariant to certain transformations in the feature space. For example, the method proposed in [**P5**] is designed to be invariant to rotation, translation and uniform scaling.

Chapter 4

Summary of the Publications

IN **the first paper** [**P1**], we proposed a fast agglomerative clustering method using a k-nearest neighbor graph. The graph is used to restrict the search of the cluster pair with the smallest agglomerative merge distance. Agglomerative clustering is a well known method for its ease of implementation and quality in terms of MSE. Unfortunately, the original method is slow, $O(N^3)$, which has later been lowered to $O(\tau N^2)$. In this work, we managed to break the $O(N^2)$ time complexity barrier by introducing an approximation to the search of the minimum merge cost. The time complexity of the algorithm is improved from $O(\tau N^2)$ to $O(\tau N \log N)$ at the cost of a slight increase in distortion. Here, τ denotes the number of nearest neighbor updates required at each iteration. According to our experiments, a relatively small neighborhood size is sufficient to maintain the quality close to that of the full search.

In the second paper [**P2**], we apply the k-nearest neighbour graph from [**P1**] to unsupervised outlier detection. We define an observation to be an outlier if, in a directed k-nearest neighbour graph, it is not a neighbour of other observations. We also propose a modification to the existing kNN distance -based method, so that user does not need to know in advance the number of outliers present in the data set. We compared the methods with real and synthetic data sets. The results show that the proposed method achieves reasonable results with synthetic data and outperforms comparative methods with real data sets when the input data set is small and performs comparatively for large datasets.

In the third paper [**P3**], we extend the k-means algorithm to robust mean vector estimation, and incorporate outlier removal into the process. In the case of overlapping symmetric mono modal distributions, k-means will include bias to the estimate of the mean. In this paper, we consider the overlapping areas of the distributions

as noise (or outliers), which is removed from the clustering process step by step. We present an outlier removal clustering (ORC) algorithm that achieves the above mentioned goal. The method employs both the clustering and outlier discovery operations to improve the estimation accuracy of the centroids of the generative distribution. The ORC algorithm consists of two stages. The first stage performs pure k-means algorithm, while the second stage iteratively removes vectors that are far from their cluster centroids. We ran a set of experiments on three synthetic data sets and three map images that were corrupted by lossy compression. The results indicate that the proposed method has a lower estimation bias on data sets with overlapping clusters than the compared methods.

In the fourth paper [**P4**], we formulate the speaker modeling task as a maximum *a posteriori* (MAP) adaptation of the VQ model. We rigorously formulate MAP adaptation for the VQ when the speaker model is adapted from UBM generated by any clustering algorithm. We show experimentally that VQ is comparable to GMM in terms of verification accuracy. Furthermore, the proposed speaker adaptation can be up to 20 times faster when using VQ than in the case of GMM.

In the fifth paper [**P5**], we design a speaker recognition system that tries to overcome technical mismatches between training and testing conditions. We propose a matching scheme, which is invariant to feature rotation, translation and uniform scaling. Channel compensation methods, such as eigenchannel [96] and NAP [19], need external training material to learn the channel factors. Whereas, we attempt to design a mismatch invariant system that operates without learning channel factors from the external training material. The proposed approach uses a neighborhood graph to represent the global shape of the feature distribution. The reference and test graphs are aligned by graph matching and the match score is computed using conventional template matching. Experiments on the NIST-1999 SRE corpus indicate that the method is comparable to the conventional GMM- and VQ-based approaches.

In the sixth paper [**P6**], we concentrate on the feature extraction part of the speaker recognition system. Most speaker recognition systems use the mel-frequency cepstral coefficients (MFCCs) to describe the spectral properties of speakers. In forensic phonetics, the long-term average spectrum (LTAS) has been used for the same purpose. It provides an intuitive graphical representation, which can be used for visualizing and quantifying differences between speakers. However, few studies have reported the use of LTAS in *automatic* speaker recognition. Thus, the purpose of the paper is to systematically study how to use the LTAS in automatic speaker recognition. We found out that it provides only marginal additional discriminative information in respect to the MFCC-based system. However, LTAS is very simple

to implement and it is order of magnitude faster in matching than MFCC-based system.

In the seventh paper [**P7**], importance of voice activity detection (VAD) for the performance of speaker recognition system is studied. We propose an improved voice activity detector based on periodicity information. A design requirement of the voice activity detector is that it needs to work in realtime environment, and to make speech/non-speech decisions without processing delay. Performance of the proposed method is compared against two existing methods: a realtime method based on long-term spectral divergence (LTSD) and a simple energy based method. The periodicity-based method outperforms the other realtime method (LTSD), and performs comparably with the energy-based method when applied for NIST 2001 and 2006 speaker recognition evaluation corpora. The method was also tested for speech-non-speech segmentation on surveillance, voice dialog and forensic recordings.

The contributions of the author of this thesis can be briefly summarized as follows. In [**P1**], the author designed, implemented and tested the first version of the method. In [**P2**, **P7**], author designed the algorithm, conducted most of the experiments and wrote the paper. In [**P3**, **P4**, **P5**], author was the main responsible of the algorithm development, implementation, writing the paper, and also participated in the experimentation. In [**P6**], author contributed in running the experiments and writing of the paper.

Chapter 5

Summary of the Results

IN this chapter, main results of the original publications [**P1**]-[**P7**] are summarized. Each paper included in this thesis contribute to some part of the general speaker recognition system. The proper place of each of the paper in the complete system is shown in Fig. 5.1. In this Figure, papers are denoted by [**P1**]-[**P7**]. Majority of the contributions are focused on the speaker modeling by cluster analysis [**P1**]-[**P4**], but contributions are also given in the voice activity detection [**P7**], feature extraction [**P6**] and pattern matching [**P5**] subsystems.

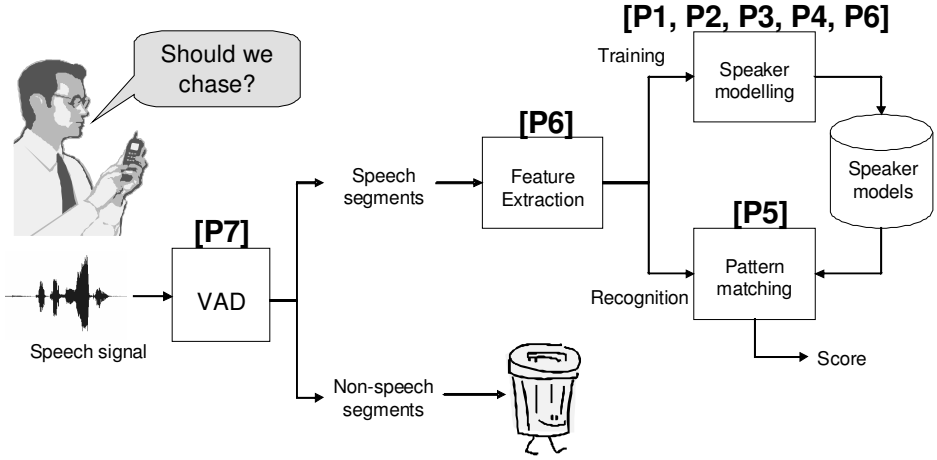


Figure 5.1: Original publications in the speaker recognition system.

Summarized results

P1: Over 120 times faster and the same clustering performance for BIRCH2 data set with DLA+k-means than with the PNN+k-means algorithm.

P2: Zero detection error with the ODIN method when applied to HR, NHL1 and NHL2 data sets and comparable performance with intrusion detection data set (KDD).

P3: About 1.5 - 2 times more accurate centroid estimation with the ORC method than with the baseline k-means.

P4: Twenty times faster speaker adaptation speed with NIST 2006 corpus using VQ-MAP than using GMM-MAP, while having exactly the same recognition accuracy (17% *equal error rate* (EER)).

P5: About 13% relative improvement in speaker identification performance when graph matching is fused with the GMM-MAP system.

P6: Over 300 times faster speaker identification with LTAS than with MFCC features. About 4% relative improvement in speaker verification accuracy when LTAS is fused with MFCC than for MFCC alone.

P7: About 63% relative improvement in speaker verification with NIST 2006 corpus if using VAD in comparison of not using VAD, and 53% relative improvement of using period VAD than using LTSD VAD.

Speech Corpora

In the speaker verification experiments, four different telephone corpora were used. All corpora were originally used as a part of the annual *National institute of standards and technology* (NIST) speaker recognition evaluations[1] . All data are conversational speech, with the sampling rate of 8 kHz and quantization 8-bit μ-law, which is a commonly used bit quantization technique in telecommunications. Corpora are summarized in Table 5.1. Difficulty of the evaluation has been increased after each new iteration of the competition.

[1] http://nist.gov/speech/tests/spk/

Table 5.1: Summary of the speech corpora

Description	Subset of NIST-1999 (I)	Subset of NIST-1999 (II)	NIST-2001	NIST-2006
Language	English	English	English	Multiple
Speakers	80	230	174	816
No. trials	950	52900	22418	53966
Telephone type	Land-line	Land-line	Mobile	Mobile
Handset mismatch	No	No	Yes	Yes
Non-speech removed	Yes	Yes	Some	No
Train speech	60 sec.	60 sec.	2 min.	5 min.
Test speech	60 sec.	60 sec.	30 sec.	5 min.
Publications where used	[**P5**]	[**P6**]	[**P4**, **P6**, **P7**]	[**P4**, **P7**]

Chapter 6

Conclusions

A PRACTICAL speaker recognition system consists of signal processing, modeling, matching and decision logic modules. In this thesis, we have studied all but decision logic, with special concentration on speaker modeling and matching based on vector quantization. One of our contributions is the formalization of a maximum *a posteriori* (MAP) vector quantization matching system using a universal background model. It achieves similar verification accuracy as Gaussian mixture model based systems but the adaptation is 20 times faster.

In our MAP adapted VQ system, the universal background model is estimated by a clustering algorithm that optimizes the model based on the squared error criterion. We gave contributions to UBM construction by designing a novel sub-quadratic time agglomerative clustering algorithm. We also contributed to the robust estimation of cluster parameters by outlier removal, first by using the k-nearest neighbour graph, and second by using a modified k-means algorithm.

Feature extraction issues were studied by experimenting on a long-term average spectrum feature with different matching techniques. We found out that LTAS is only marginally useful when combined with state-of-the-art GMM-MFCC system, but it is computationally efficient. We proposed a new realtime voice activity detection method for speaker recognition, and experimented with the effect of the voice activity parameters on the speaker recognition accuracy. Contrary to the earlier reports, we found out that the accuracy of the whole system is greatly affected by proper tuning of the VAD parameters. We noticed that the voice activity detector should be tuned so that as few non-speech frames are accepted as possible.

The performance of the speaker recognition system is affected by any kind of mismatch between the audio material used to train the speaker model and the test utterance. Mismatch can happen, for example, when transmission channel (microphone or audio coding system), language or emotion is different. In this thesis, we

contributed to the field of mismatch compensation by designing a restricted affine invariant modeling and matching scheme using the graph matching methodology.

As future work, one should study different ways to overcome exponential time complexity of the graph matching approach. By using a graph matching algorithm based on the *spectral graph theory* one could apply linear algebra techniques instead of combinatorial optimization methods. We would like to improve on the combinatorial matching algorithm where matching is now done by reduction to a clique problem and then solving the clique by a local search algorithm. Direct local search solution makes it possible to use larger graphs in matching. Another direction of future research would be to match SVMs using the MAP adapted vector quantization models. Eigenchannel channel compensation technique, originally proposed for the Gaussian mixture models, could be adapted to the vector quantization framework as well.

References

[1] Ackermann, M. R., Blömer, J., and Sohler, C. Clustering for metric and non-metric distance measures. In *ACM-SIAM Symposium on Discrete Algorithms (SODA 2008)* (San Francisco, California, January 2008), pp. 799–808.

[2] Arora, S., and Kannan, R. Learning mixtures of arbitrary Gaussians. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC 2001)* (2001), pp. 247–257.

[3] Arthur, D., and Vassilvistkii, S. How slow is the k-means method? In *Proceedings of the 22nd Annual Symposium on Computational Geometry (SCG 2006)* (June 2006), pp. 144–153.

[4] Arthur, D., and Vassilvitskii, S. k-means++: the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms (SODA 2007)* (New Orleans, Louisiana, 2007), pp. 1027–1035.

[5] Asano, T., Battacharya, B., Keil, M., and Yao, F. Clustering algorithms based on minimum and maximum spanning trees. In *Proceedings of the 4th Annual Symposium on Computational Geometry (SCG 1988)* (1988), pp. 252–257.

[6] Atal, B. Automatic speaker recognition based on pitch contours. *Journal of Acoustic Society of America 52*, 6 (1972), 1687–1697.

[7] Auckenthaler, R., Carey, M., and Lloyd-Thomas, H. Score normalization for text-independent speaker verification systems. *Digital Signal Processing 10* (2000), 42–54.

[8] Aurenhammer, F. Voronoi diagrams – a survey of a fundamental geometric data structure. *ACM Computing Surveys 23*, 3 (1991), 345–405.

[9] Bay, S. D., and Schwabacher, M. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD 2003)* (Washington, D.C., 2003), pp. 29–38.

[10] Benyassine, A., Schlomot, E., and Su, H. ITU-T recommendation g729 annex b: A silence compression scheme for use with g729 optimized for v.70

digital simultaneous voice and data applications. *IEEE Communications Magazine 35* (1997), 64–73.

[11] Bimbot, F., Bonastre, J.-F., Fredouille, C., Gravier, G., Meignier, S., Merlin, T., Ortega-Garcia, J., Magrin-Chagnolleau, I., Petrovska-Delacretaz, D., and Reynolds, D. A. A tutorial on text-independent speaker verification. *EURASIP Journal on Applied Signal Processing*, 4 (2004), 430–451.

[12] Bishop, C. M. *Pattern Recognition and Machine Learning*. Springer-Verlag, New York, 2006.

[13] Boberg, J., and Salakoski, T. General formulation and evaluation of agglomerative clustering methods with metric and non-metric distances. *Pattern Recognition 26*, 9 (September 1993), 1395–1406.

[14] Bottou, L., and Bengio, Y. Convergence properties of the k-means algorithms. In *Advances in Neural Information Processing Systems (NIPS 1995)* (1995), pp. 585–592.

[15] Brandes, U., Delling, D., Gaertler, M., Görke, R., Hoefer, M., Nikolski, Z., and Wagner, D. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering 20*, 2 (February 2008), 172–188.

[16] Breunig, M. M., Kriegel, H. P., Ng, R. T., and Sanders, J. LOF: Identifying density based local outliers. In *Proceedings of the 19th International Conference on Management of Data (SIGMOD 2000)* (Dallas, Texas, 2000).

[17] Brito, M. R., Chávez, E. L., Quiroz, A. J., and Yukich, J. E. Connectivity of the mutual k-nearest-neighbor graph in clustering and outlier detection. *Statistics & Probability Letters 35*, 1 (August 1997), 33–42.

[18] Brusco, M. J. A repetitive branch-and-bound procedure for minimum within-cluster sums of squares partitioning. *Psychometrika 71*, 2 (June 2006), 347–363.

[19] Burget, L., Pavel Matejka, M., Schwarz, P., Glembek, O., and Cernocký, J. Analysis of feature extraction and channel compensation in a GMM speaker recognition system. *IEEE Transactions on Audio, Speech, and Language Processing 17*, 7 (September 2007), 1979–1986.

[20] Callahan, P. B., and Kosaraju, S. R. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *Journal of the Association for Computing Machinery 42*, 1 (1995), 67–90.

[21] Campbell, W., Campbell, J., Reynolds, D., Singer, E., and Torres-Carrasquillo, P. Support vector machines for speaker and language recognition. *Computer Speech & Language 20*, 2–3 (April–July 2006), 210–229.

[22] Campbell, W. M., Sturim, D. E., and Reynolds, D. A. Support vector machines using GMM supervectors for speaker verification. *IEEE Signal Processing Letters 13*, 5 (2006), 308–311.

[23] Cao, L. J., Lee, H. P., and Chong, W. K. Modified support vector novelty detector using training data from outliers. *Pattern Recognition Letters 24* (2003), 247–2487.

[24] Capoyleas, V., Rote, G., and Woeginger, G. Geometric clustering. *Journal of Algorithms 12* (1991), 341–356.

[25] Cardinal, J., and Eppstein, D. Lazy algorithms for dynamic closest pair with arbitrary distance measures. In *Proceedings of the SIAM Workshop on Algorithm Engineering and Experiments (ALENEX04)* (New Orleans, January 2004), pp. 112–119.

[26] Celeux, G., Chretien, S., Forbes, F., and Mkhadri, A. A component-wise EM algorithm for mixtures. *Journal of Computational and Graphical Statistics 10*, 4 (December 2001), 697–712.

[27] Chávez, E. A subquadratic algorithm for clustering and outlier detection in massive metric data. In *Proceedings of the 8th International Symposium on String Processing and Information Retrieval (SPIRE 2001)* (Laguna de San Rafael, Chile, November 2001), pp. 46–58.

[28] Clarkson, K. L. Fast algorithms for the all-nearest-neighbor problem. In *Proceedings 24th IEEE Symposium Foundations of Computer Science (FOCS 1983)* (1983), pp. 226–232.

[29] Clarkson, K. L. Nearest-neighbor searching and metric space dimensions. In *Nearest-Neighbor Methods for Learning and Vision: Theory and Practice*, G. Shakhnarovich, T. Darrell, and P. Indyk, Eds. MIT Press, 2006, pp. 15–59.

[30] Cormen, T. H., Leiserson, C. E., and Rivest, R. L. *Intoduction to Algorithms*. The MIT Press, 1998.

[31] Dasgupta, S. Learning mixtures of Gaussians. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS 1999)* (1999), pp. 634–644.

[32] Dasgupta, S., and Schulman, L. J. A two-round variant of EM for Gaussian mixtures. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI 2000)* (2000), pp. 152–159.

[33] Davis, S., and Mermelstein, P. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing 28*, 4 (1980), 357–366.

[34] Day, W. H. E. Complexity theory: An introduction for practitioners of classification. In *Clustering and Classification*, P. Arabie, L. Hubert, and G. De Soete, Eds. World Scientific Publishing Co., Singapore, 1992, pp. 199–233.

[35] De Loera, J. A., Hemmecke, R., Onn, S., Rothblum, U. G., and Weismantel, R. Convex integer maximization via graver bases. *Journal of Pure and Applied Algebra* (2008). to appear.

[36] Dehak, N., Dumouchel, P., and Kenny, P. Modeling prosodic features

with joint factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing 15*, 7 (September 2007), 2095–2103.

[37] Dempster, A., Laird, N., and Rubin, D. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B 39* (1977), 1–38.

[38] Drineas, P., Frieze, A., Kannan, R., Vempala, S., and Vinay, V. Clustering large graphs via the singular value decomposition. *Machine Learning 56*, 1–3 (July 2004), 9–33.

[39] Du, Q., Faber, V., and Gunzburger, M. Centroidal Voronoi tessellations: Applications and algorithms. *SIAM Review 41*, 4 (December 1999), 637–676.

[40] Dunford, N., and Schwartz, J. T. *Linear operators*, vol. I. Wiley-Interscience, 1958.

[41] Easter, M., Kriegel, H. P., Sanders, J., and Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD 1996)* (1996), pp. 226–231.

[42] El-Hamdouchi, A., and Willett, P. Hierarchic document classification using ward's clustering method. In *Proceedings of the 9th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 1986)* (New York, NY, USA, 1986), pp. 149–156.

[43] Elkan, C. Using the triangle inequality to accelerate k-means. In *Proceedings of the 20th International Conference on Machine Learning (ICML 2003)* (Washington DC, USA, 2003), pp. 147–153.

[44] Eppstein, D. Fast hierarchical clustering and other applications of dynamic closest pair. *The ACM Journal of Experimental Algorithmics 5*, 1 (June 2000), 1–23.

[45] Eppstein, D. Squarepants in a tree: sum of subtree clustering and hyperbolic pants decomposition. In *Proceedings of the 18th annual ACM-SIAM symposium on Discrete algorithms (SODA 2007)* (Philadelphia, PA, USA, 2007), pp. 29–38.

[46] Eppstein, D., Paterson, M. S., and Yao, F. F. On nearest-neighbor graphs. *Discrete and Computational Geometry 17* (1997), 263–282.

[47] Equitz, W. H. A new vector quantization clustering algorithm. *IEEE Transactions on Acoustics, Speech and Signal processing 37*, 10 (October 1989), 1568–1575.

[48] Farrel, K., Mammone, R., and Assaleh, K. Speaker recognition using neural networks and conventional classifiers. *IEEE Transactions on Speech and Audio Processing 2*, 1 (1994), 194–205.

[49] Feldman, D., Monemizadeh, M., and Sohler, C. A PTAS for k-means clustering based on weak coresets. In *Proceedings of the 23rd annual symposium on Computational geometry (SCG 2007)* (New York, NY, USA, 2007),

ACM, pp. 11–18.

[50] Fessler, J. A., and Hero, A. O. Space-alternating generalized expectation-maximization algorithm. *IEEE Transactions on Signal Processing 42*, 10 (October 1994), 2664–2677.

[51] Figueiredo, M., and Jain, A. Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*, 3 (March 2002), 1–16.

[52] Fränti, P., Kaukoranta, T., Shen, D. F., and Chang, K.-S. Fast and memory efficient implementation of the exact pnn. *IEEE Transactions on Image Processing 9*, 5 (2000), 358–369.

[53] Fränti, P., and Kivijärvi, J. Randomized local search algorithm for the clustering problem. *Pattern Analysis & Applications 3*, 4 (2000), 358–369.

[54] Fränti, P., Virmajoki, O., and Hautamäki, V. Fast agglomerative clustering using a k-nearest neighbour graph. *IEEE Transactions on Pattern Analysis and Machine Intelligence 28*, 11 (November 2006), 1875–1881.

[55] Fränti, P., Virmajoki, O., and Kaukoranta, T. Branch-and-bound technique for solving optimal clustering. In *Proceedings of the 16th International Conference on Pattern Recogntion (ICPR 2002)* (Quebec, Canada, August 2002), pp. 232–235.

[56] Freedman, D., Purves, R., and Pisani, R. *Statistics*. W.W. Norton, New York, 1978.

[57] Friedman, J. H., Bentley, J. L., and Finkel, R. A. An algorithm for finding best matches in logarithmic exptected time. *ACM Transactions on Mathematical Software 3*, 3 (1977), 209–226.

[58] Garey, M. R., and Johnson, D. S. *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman, San Francisco, 1979.

[59] Gauvain, J. L., and Lee, C.-H. Maximum *a posteriori* estimation for multivariate Gaussian mixture observations of Markov chain. *IEEE Transactions on Speech and Audio Processing 2*, 2 (April 1994), 291–298.

[60] Gersho, A., and Gray, R. M. *Vector Quantization and Signal Compression*. Kluwer Academic Publisher, Boston, USA, 1992.

[61] Ghoting, A., and Parthasarathy, S. Knowledge-conscious exploratory data clustering. In *Proceedings of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2006)* (Berlin, Germany, September 2006), pp. 511–519.

[62] Gower, J. C., and Ross, G. I. Maximum spanning trees and single linkage cluster analysis. *Applied Statistics 18*, 1 (1969), 54–64.

[63] Guha, S., Rastogi, R., and Shim, K. CURE an efficient clustering algorithm for large databases. In *Proceedings of the 17th International Conference on Management of Data (SIGMOD 1998)* (Seattle, Washington, June 1998), pp. 73–84.

[64] Guha, S., Rastogi, R., and Shim, K. ROCK: A robust clustering algorithm for categorical attributes. In *Proceedings of the 15th International Conference on Data Engineering (ICDE 1999)* (1999), pp. 512–521.

[65] Hansen, P., and Delattre, M. Complete-link cluster analysis by graph coloring. *Journal of American Statistical Association 73*, 362 (June 1978), 397–403.

[66] Hansen, P., and Jaumard, B. Cluster analysis and mathematical programming. *Mathematical Programming 79*, 1–3 (October 1997), 191–215.

[67] Har-Peled, S., and Sadri, B. How fast is the k-means method? *Algorithmica 41* (2005), 185–202.

[68] Harmegnies, B. SDDD: A new dissimilarity index for the comparison of speech spectra. *Pattern Recognition Letters 8*, 3 (October 1988), 153–158.

[69] Harrington, J., and Cassidy, S. *Techniques in Speech Acoustics*. Kluwer Academic Publishers, Dordrecth, 1999.

[70] Hartigan, J. A. *Clustering Algorithms*. Wiley, New York, 1975.

[71] Hautamäki, V., Cherednichenko, S., Kärkkäinen, I., Kinnunen, T., and Fränti, P. Improving k-means by outlier removal. In *Proceedings of the 14th Scandinavian Conference on Image Analysis (SCIA 2005)* (Joensuu, Finland, June 2005), pp. 978–987.

[72] Hautamäki, V., Kärkkäinen, I., and Fränti, P. Outlier detection using k-nearest neighbour graph. In *International Conference on Pattern Recognition (ICPR 2004)* (Cambridge, UK, August 2004), vol. 3, pp. 430–433.

[73] Hautamäki, V., Kinnunen, T., and Fränti, P. Text-independent speaker recognition using graph matching. *Pattern Recogntion Letters 29*, 9 (July 2008), 1427–1432.

[74] Hautamäki, V., Kinnunen, T., Kärkkäinen, I., Saastamoinen, J., Tuononen, M., and Fränti, P. Maximum *a posteriori* adaptation of the centroid model for speaker verification. *IEEE Signal Processing Letters 15* (2008), 162–165.

[75] Hautamäki, V., Tuononen, M., Niemi-Laitinen, T., and Fränti, P. Improving speaker verification by periodicity based voice activity detection. In *Proceedings of the 12th International Conference on Speech and Computer (SPECOM 2007)* (Moscow, Russia, October 2007), vol. 2, pp. 645–650.

[76] Hawkins, D. M. *Identification of Outliers*. Chapman and Hall, London, 1980.

[77] He, Y., and Chen, L. MinClue: A MST-based clustering method with auto-threshold-detection. In *IEEE Conference on Cybernetics and Intelligent Systems* (Singapore, December 2004), pp. 229–233.

[78] Hochbaum, D. S. When are NP-hard location problems easy? *Annals of Operations Research 1* (1984), 201–214.

[79] Hollien, H., and Majewski, W. Speaker identification by long-term spec-

tra under normal and distorted speech conditions. *Journal of the Acoustical Society of America 62*, 4 (October 1977), 975–980.

[80] Huijbregts, M., and Wooters, C. The blame game: Performance analysis of speaker diarization system components. In *Proceedings of 8th International Interspeech Event (Interspeech 2007)* (August 2007), pp. 1857–1860.

[81] Hwang, F. K., Onn, S., and Rothblum, U. G. A polynomial time algorithm for shaped partition problems. *SIAM Journal of Optimization 10* (1999), 70–81.

[82] Inaba, M., Katoh, N., and Imai, H. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering. In *Proceedings of the 10th Annual ACM symposium on computational geometry (SCG 1994)* (1994), pp. 332–339.

[83] Jain, A. K., and Dubes, R. C. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[84] Jarvis, R. A., and Patrick, E. A. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on Computers C-22*, 11 (November 1973), 1025–1034.

[85] Jensen, R. A dynamic programmig algorithm for cluster analysis. *Operations Research 17*, 6 (November–December 1969), 1034–1057.

[86] Jiang, M. F., Tseng, S. S., and Su, C. M. Two-phase clustering process for outliers detection. *Pattern Recognition Letters 22* (2001), 691–700.

[87] Jin, W., Tung, A. K. H., and Han, J. Finding top-n local outliers in large database. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2001)* (2001), pp. 293–298.

[88] Johnson, S. C. Hierarchical clustering schemes. *Psychometrika 32*, 3 (September 1979), 241–254.

[89] Kang, S.-I., Jo, Q.-H., and Chang, J.-H. Discriminative weight training for statistical model-based voice activity detection. *IEEE Signal Processing Letters 15* (2008), 170–173.

[90] Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., and Wu, W. Y. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24* (2002), 881–892.

[91] Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., and Wu, W. Y. A local search approximation algorithm for k-means clustering. *Computational Geometry 28* (2004), 89–112.

[92] Karypsis, G., Han, E.-H., and Kumar, V. CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *IEEE Computer 32*, 8 (August 1999), 68–75.

[93] Kaufman, L., and Rousseeuw, P. J. *Finding groups in data: an introduction to cluster analysis*. John Wiley Sons, New York, 1990.

[94] Kaukoranta, T., Fränti, P., and Nevalainen, O. Vector quantization by lazy pairwise nearest neighbor method. *Optical Engineering 38*, 11 (November 1999), 1862–1868.

[95] Kaukoranta, T., Fränti, P., and Nevalainen, O. A fast GLA based on code vector activity detection. *IEEE Transactions on Image Processing 9*, 8 (August 2000), 1337–1342.

[96] Kenny, P., Boulianne, G., Ouellet, P., and Dumouchel, P. Speaker and session variability in GMM-based speaker verification. *IEEE Transactions on Audio, Speech and Language Processing 15*, 4 (May 2007), 1448–1460.

[97] Kinnunen, T. *Spectral Features for Automatic Text-Independent Speaker Recognition.* Licentiate's thesis, University of Joensuu, Department of Computer Science, Joensuu, Finland, 2004.

[98] Kinnunen, T. *Optimizing Spectral Feature Based Text-independent Speaker Recognition.* PhD thesis, University of Joensuu, 2005.

[99] Kinnunen, T., and González Hautamäki, R. Long-term F0 modeling for text-independent speaker recognition. In *Proceedings of the 10th International Conference Speech and Computer (SPECOM 2005)* (Patras, Greece, October 2005), pp. 567–570.

[100] Kinnunen, T., Hautamäki, V., and Fränti, P. On the use of long-term average spectrum in automatic speaker recogntion. In *Proceedings of the 5th International Symposium on Chinese Spoken Language Processing (ISCSLP 2006)* (Singapore, December 2006), vol. 2, pp. 559–567.

[101] Kinnunen, T., Karpov, E., and Fränti, P. Real-time speaker identification and verification. *IEEE Transactions on Audio, Speech and Language Processing 14*, 1 (January 2006), 277–288.

[102] Kinnunen, T., Kilpeläinen, T., and Fränti, P. Comparison of clustering algorithms in speaker identification. In *Proceedings of the IASTED International Conference on Signal Processing and Communications (SPC 2000)* (Marbella, Spain, September 19-22 2000), pp. 222–227.

[103] Kinnunen, T., Tuononen, M., and Fränti, P. Which clustering algorithm to select for text-independent speaker recognition? *Pattern Recognition* (2008). Submitted.

[104] Kinoshita, Y., Ishihara, S., and Rose, P. Beyond the long-term mean: Exploring the potential of F0 distribution parameters in traditional forensic speaker recognition. In *Proceesings of the Odyssey: The Speaker and Language Recognition Workshop* (Stellenbosch, South Africa, January 2008).

[105] Knorr, E. M., and Ng, R. T. Unified notion of outliers: Properties and computation. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD 1997)* (Newport Beach, CA, 1997), pp. 219–222.

[106] Knorr, E. M., and Ng, R. T. Algorithms for mining distance-based outliers

in large datasets. In *Proceedings 24th International Conference Very Large Data Bases (VLDB 1998)* (New York, USA, 1998), pp. 392–403.

[107] Koontz, W. L. G., Nareda, P. M., and Fukunaga, K. A branch and bound clustering algorithm. *IEEE Transactions on Computers 24*, 9 (September 1975), 908–915.

[108] Kou, Y., Lu, C.-T., and Chen, D. Spatial weighted outlier detection. In *Proceedings of the 5th SIAM International Conference on Data Mining (SDM 2006)* (2006).

[109] Krishnapuram, R., and Keller, J. A possibilistic approach to clustering. *IEEE Transactions on Fuzzy Systems 1*, 2 (1993), 98–110.

[110] Kuncheva, L. *Fuzzy Classifier Design*. Physica-Verlag, 2000.

[111] Kurita, T. An efficient agglomerative clustering using a heap. *Pattern Recognition 25*, 3 (March 1991), 205–209.

[112] Lee, W., Stolfo, S. J., and Mok, K. W. A data mining framework for building intrusion detection models. In *Proceedings of the IEEE Symposium on Security and Privacy* (May 1999), pp. 120–132.

[113] Lee, Y., Seo, C., Lee, J., and Lee, K. Y. Speaker verification system for PDA in mobile-commerce. In *Proceedings of the 2nd International Conference on Human Society@Internet (HSI 2003)* (Seoul, Korea, June 2003), pp. 668–674.

[114] Liao, Y., and Vemuri, V. R. Use of k-nearest neighbor classifier for intrusion detection. *Computers & Security 21*, 5 (2002), 439–448.

[115] Linde, Y., Buzo, A., and Gray, R. M. An algorithm for vector quantizer desing. *IEEE Transactions 28*, 1 (January 1980), 84–95.

[116] Lindh, J. Visual acoustic vs. aural perceptual speaker identification in closed set of disguised voices. In *Proceedings of The 18th Swedish Phonetics Conference (FONETIK 2005)* (2005), pp. 17–20.

[117] Liu, J. Neural networks with enhanced outlier rejection ability for off-line handwritten word recognition. *Pattern Recognition 35* (2002), 2061–2071.

[118] Liu, J., and Gader, P. Outlier rejection with MLPs and variants of RBF networks. In *Proceedings of The 15th International Conference on Pattern Recognition (ICPR 2000)* (2000), pp. 680–683.

[119] Liu, X., Cheng, G., and Wu, J. X. Analyzing outliers cautiously. *IEEE Transactions on Knowledge and Data Engineering 14*, 2 (March/April 2002), 432–437.

[120] Luoma, O., Tuikkala, J., and Nevalainen, O. Accelerating GLA with an M-tree. In *Proceedings of The Second World Enformatika Conference (WEC 2005)* (Turkey, February 2005), pp. 196–199.

[121] M. R. Garey, D. S. J., and Witsenhausen, H. S. The complexity of the generalized lloyd-max problem. *Transactions on Information Theory 28*, 2 (1982), 255–256.

[122] Ma, J., and Perkins, S. Time-series novelty detection using one-class support vector machines. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2003)* (October 2003), pp. 1741–1745.

[123] MacQueen, J. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability* (University of California, 1967), vol. I, pp. 281–297.

[124] Mak, M.-W., and Tsang, C.-L. Stochastic feature transformation with divergence-based out-of-handset rejection for robust speaker verification. *EURASIP Journal on Applied Signal Processing 4* (January 2004), 452–465.

[125] Mammone, R. J., Zhang, X., and Ramachandran, R. P. Robust speaker recognition: A feature-based approach. *IEEE Signal Processing Magazine 13*, 5 (September 1996), 58–71.

[126] Markel, J., Oshika, B., and Gray, A. H. Long-term feature averaging for speaker recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing 25*, 4 (1977), 330–337.

[127] Markou, M., and Singh, S. Novelty detection: A review. *Signal Processing 83*, 12 (December 2003), 2481 – 2497.

[128] Martínez, A. M., and Vitriàc, J. Learning mixture models using a genetic version of the EM algorithm. *Pattern Recognition Letters 21*, 8 (July 2000), 759–769.

[129] Matousek, J. On approximate geometric k-clustering. *Discrete Computational Geometry 24* (2000), 61–84.

[130] McKenzie, P., and Alder, M. Initializing the EM algorithm for use in Gaussian mixture modelling. Tech. rep., University of Western Australia, 1993.

[131] Meilă, M. The uniqueness of a good optimum for k-means. In *Proceedings of the 23rd International Conference on Machine Learning (ICML 2006)* (Pittsburgh, PA, USA, 2006), pp. 625–632.

[132] Merle, O. D., Hansen, P., Jaumard, B., and Mladenović, N. An interior point algorithm for minimum sum-of-squares clustering. *SIAM Journal of Scientific Computing*, 4 (2000), 1485–1505.

[133] Minigawa, A., Tagawa, N., and Tanaka, T. SMEM algorithm is not fully compatible with maximum-likelihood framework. *Neural Computation 14* (2002), 1261–1266.

[134] Moore, A. The anchors hierarchy: Using the triangle inequality to survive high-dimensional data. In *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence (UAI 2000)* (San Francisco, CA, USA, 2000), pp. 397–405.

[135] Murtagh, F. A survey of recent advances in hierarchical clustering algorithms. *The Computer Journal 26*, 4 (1983), 354–359.

[136] Oppenheim, A., and Schafer, R. *Digital Signal Processing*. Prentice Hall,

1975.

[137] Ostrovsky, R., Rabani, Y., Schulman, L. J., and Swamy, C. The effectiveness of Lloyd-type methods for the k-means problem. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006)* (2006), pp. 165–176.

[138] Päivinen, N. Clustering with a minimum spanning tree of scale-free-like structure. *Pattern Recognition Letters 26*, 7 (2005), 921–930.

[139] Papadimitrou, C. Worst-case and probabilistic analysis of a geometric location problems. *SIAM Journal on Computing* (1981).

[140] Patcha, A., and Park, J.-M. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks 51* (2007), 3448–3470.

[141] Pavan, M., and Pelillo, M. Dominant sets and pairwise clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence 29*, 1 (2007), 167–172.

[142] Peña, J. M., Lozano, J. A., and Larrañaga, P. An empirical comparison of four inititialization methods for the k-means algorithm. *Pattern Recognition Letters*, 10 (October 1999), 1027–1040.

[143] Pelecanos, J., and Sridharan, S. Feature warping for robust speaker verification. In *Proceedings of the Speaker Odyssey 2001* (Crete, Greece, 2001), pp. 213–218.

[144] Pelleg, D., and Moore, A. W. Accelerating exact k-means algorithms with geometric reasoning. In *Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (KDD 1999)* (San Diego, California, USA, 1999), pp. 277–281.

[145] Peng, J., and Xia, Y. A cutting algorithm for the minimum sum-of-squared error clustering. In *Proceedings of the fifth SIAM International Conference on Data Mining (SDM 2005)* (Newport Beach, California, April 2005), pp. 150–160.

[146] Pernkopf, F., and Bouchffra, D. Genetic-based EM algorithm for learning Gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27*, 8 (2005), 1344–1348.

[147] Ramaswamy, S., Rastogi, R., and Shim, K. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 19th International Conference on Management of Data (SIGMOD 2000)* (Dallas, Texas, May 2000), pp. 427–438.

[148] Ramírez, J., Segura, J., Benítez, C., de la Torre, A., and Rubio, A. Efficient voice activity detection algorithms using long-term speech information. *Speech Communication 42* (2004), 271–287.

[149] Rand, W. M. Objective criteria for the evaluation of clustering methods. *Journal of American Statistical Association 66*, 336 (December 1971), 846–

850.

[150] Rao, M. R. Cluster analysis and mathematical programming. *Journal of the American Statistical Association 66*, 335 (September 1971), 622–626.

[151] Reynolds, D., Andrews, W., Campbell, J., Navratil, J., Peskin, B., Adami, A., Jin, Q., Klusacek, D., Abramson, J., J., M. R. G., Jones, D., and Xiang, B. The SuperSID project: exploiting high-level information for high-accuracy speaker recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP 2003)* (Hong Kong, 2003), pp. 784–787.

[152] Reynolds, D., Quatieri, T., and Dunn, R. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing 10*, 1 (2000), 19–41.

[153] Reynolds, D., and Rose, R. Robust text-independent speaker identification using Gaussian mixture models. *IEEE Transactions on Speech and Audio Processing 3* (1995), 72–83.

[154] Rodríguez Liñares, L., García Mateo, C., and Alba Castro, J. On combining classifiers for speaker authentication. *Pattern Recognition 36* (2003), 347–359.

[155] Rose, P. *Forensic Speaker Identification*. Taylor & Francis, London, 2002.

[156] Sambur, M. Selection of acoustic features for speaker identification. *IEEE trans. on Acoustics, Speech, and Singal Processing 23*, 2 (April 1975), 176–182.

[157] Schaeffer, S. E. Graph clustering. *Computer Science Review 1*, 1 (August 2007), 27–64.

[158] Shanbehzadeh, J., and Ogunbona, P. O. On the computational complexity of the LBG and PNN algorithms. *IEEE Transactions on Image Processing 6* (1997), 614–616.

[159] Siohan, O., and Lee, C.-H. Iterative noise and channel estimation under the stochastic matching algortihm framework. *IEEE Signal Processing Letters 4*, 11 (November 1997), 304–306.

[160] Soong, F., Rosenberg, A., Juang, B.-H., and Rabiner, L. R. A vector quantization approach to speaker recognition. *AT & T Technical Journal 66* (1987), 14–26.

[161] Späth, H. *Cluster analysis algorithms for data reduction and classification of objects*. Wiley, new York, 1980.

[162] Sperotto, A., and Pelillo, M. Szemerédi regularity lemma and its applications to pairwise clustering and segmentation. In *The 6th International Conference Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR 2007)* (EZhou, Hubei, China, August 2007), pp. 13–27.

[163] Srebro, N. Are there local maxima in the infinite-sample likelihood of Gaussian mixture estimation? In *Proceedings of the 20th Annual Conference on*

Learning Theory (COLT 2007) (San Diego, California, June 2007), pp. 628–629.

[164] Steen, L. A., and Seebach, A. *Counterexamples in Topology.* Dover publications, 1995.

[165] Steinley, D. k-means clustering: A half-century synthesis. *British journal of Mathematical and Statistical Psychology 59* (2006), 1–34.

[166] Tawfick, M. M., Abbas, H. M., and Shahein, H. I. An integer-coded evolutionary approach for mixture maximum likelihood clustering. *Pattern Recogntion Letters 29* (2008), 515–524.

[167] Tax, D. M. J., and Duin, R. P. W. Support vector data description. *Pattern Recognition Letters 20* (1999), 1191–1199.

[168] Teunen, R., Shahshahani, B., and Heck, L. A model-based transformational approach to robust speaker recognition. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP 2000)* (Beijing, China, 2000), vol. 2, pp. 495–498.

[169] Toledo-Ronen, O. Speech detection for text-dependent speaker verification. In *Proceedings of the Speaker Odyssey 2001* (Crete, Greece, June 2001), pp. 33–36.

[170] Tong, R., Ma, B., Lee, K., You, C., Zhou, D., Kinnunen, T., Sun, H., Dong, M., Chng, E., and Li, H. Fusion of acoustic and tokenization features for speaker recognition. In *Proceedings of the 5th International Symposium on Chinese Spoken Language Processing (ISCSLP 2006)* (Singapore, December 2006), pp. 494–505.

[171] Tranter, S. E., and Reynolds, D. A. An overview of automatic speaker diarization systems. *IEEE Transactions on Audio, Speech, and Language Processing 14*, 5 (September 2006), 1557–1565.

[172] Tuononen, M., González Hautamäki, R., and Fränti, P. Automatic voice activity detection in different speech applications. In *Proceedings of the International Conference on Forensic Applications and Techniques in Telecommunications, Information and Multimedia (e-Forensics 2008)* (Adelaide, Australia, January 2008).

[173] Ueda, N., Nakano, R., Ghahramani, Z., and Hinton, G. E. SMEM algorithm for mixture models. *Neural Computation 12* (2000), 2109–2128.

[174] Vaidya, P. An $o(n \log n)$ algorithm for the all-nearest-neighbors problem. *Discrete and Computational Geometry 4* (1989), 101–115.

[175] Varošanec-Škarić, G., and Bićanić, J. A comparison of indices of difference and similarity, based on LTASS and tested on voices in real forensic case and in controlled conditions. In *Proceedings of the 16th International Congress of Phonetic Sciences (ICPhS 2007)* (Saarbücken, Germany, August 2007), pp. 2085–2088.

[176] Vazirani, V. *Approximation Algorithms.* Springer, 2003.

[177] Velho, L., Gomez, J., Vicinius, M., and Sobreiro, R. Color image quantization by pairwise clustering. In *Proceedings of the Brazilian Symposium of Computer Graphics and Image Processing (SIBGRAPI 1997)* (Campos de Jordão, Brazil, October 1997), pp. 596–601.

[178] Vempala, S., and Wang, G. A spectral algorithm for learning mixture models. *Journal of Computer and System Sciences 68*, 4 (June 2003), 841–860.

[179] Verbeek, J. J., Vlassis, N., and Krose, B. Efficient greedy learning of Gaussian mixture models. *Neural Computation 15*, 2 (2003), 129–151.

[180] Virmajoki, O., Fränti, P., and Kaukoranta, T. Practical methods for speeding-up the pairwise nearest neighbor method. *Optical Engineering 40*, 11 (November 2001), 2495–2504.

[181] Vogt, R., and Sridharan, S. Explicit modeling of session variability for speaker verification. *Computer Speech & Language 22*, 1 (January 2008), 17–38.

[182] Ward, J. H. A new vector quantization clustering algorithm. *Journal of American Statistical Association 58* (1963), 236–244.

[183] Welch, P. D. The use of fast Fourier transforms for the estimation of the power spectra. *IEEE Transactions on Audio and Electroacoustics 15* (1967), 70–73.

[184] Williams, G., Baxter, R., He, H., Hawkings, S., and Gu, L. A comparative study of RNN for outlier detection in data mining. In *Proceedings of the 2nd IEEE International Conference on Data Mining (ICDM 2002)* (Maebashi City, Japan, December 2002).

[185] Xu, R., and Wunsch, D. Survey of clustering algorithms. *IEEE Transactions on Neural Networks 16*, 3 (May 2005), 645–678.

[186] Xu, Y., Olman, V., and Xu, D. Minimum spanning trees for gene expression data clustering. *Genome Informatics 12* (2001), 24–33.

[187] Yamanishi, K., Takeuchi, J., Williams, G., and Milne, P. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithm. In *Proceedings The 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2000)* (August 2000), pp. 320–324.

[188] Yin, J., Yang, Q., and jeffrey Junfeng Pan. Sensor-based abnormal human-activity detection. *IEEE Transactions on Knowledge and Data Engineering* (2008). (in press).

[189] Zhang, T., Ramakrishnan, R., and Livny, M. BIRCH: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery 1*, 2 (1997), 141–182.

[190] Zhang, Y., yang, S., and Wang, Y. Y. LDBOD: A novel local distribution based outlier detector. *Pattern Recogntion Letters 29* (May 2008), 967–976.

[191] Zhang, Z., Dai, B. T., and Tung, A. K. H. On the lower bound of local optimums in k-means algorithm. In *Proceedings of the 6th International Conference on Data Mining (ICDM 2006)* (Los Alamitos, CA, USA, 2006), pp. 775–786.

[192] Äyrämö, S. *Knowledge Mining Using Robust Clustering*. PhD thesis, University of Jyväskylä, 2006.