The University of Hong Kong

Faculty of Engineering

Department of Computer Science

COMP7704
Dissertation Title
Real-time Speaker Recognizer

Submitted in partial fulfillment of the requirements for the admission to the degree of
Master of Science in Computer Science

By
Pan Hao
3035349015

Supervisor's title and name: Dr. Beta C.L. Yip
Date of submission: 01/08/2019

Abstract

Speaker diarization has become more important in many speech processing tasks recently. Most state-of-the-art speaker diarization system decodes in an offline fashion and requires intensive computation and long processing time, which leads to the handicap for real-time applications. In this paper, we implemented the binary key speaker modelling approaches and built a fast offline speaker diarization system that can label the speakers in recorded audio, with visualization and audio player panel. An advanced version of the system is also developed, which can process the audio data in real-time with acceptable delay and correct earlier outputs when necessary.

Keywords: Speaker diarization, binary key speaker modelling, speaker clustering, MFCC

Declaration

The work contained in this dissertation has not been submitted to any publications, awards or education institutions. To the best of my knowledge and belief, this dissertation contains no material from other writers except where references are made.

Acknowledgement

Firstly, I would like to thank my supervisor, Dr. Beta C.L. Yip, for his great advice and assistance in this dissertation.

Secondly, I would also like to thank Jose Patino, one of the researchers in the area of speaker diarization and binary key speaker modelling, who replied my email to answer my questions clearly and thoroughly.

I am also grateful to my boss Abby Ye for her support and understanding. She allows me to take the annual leave several times for this project despite the busy work in the company.

Finally, I would like to thank my family. Nothing would be possible without them.

Table of Contents

List of Figures

1. Introduction

1.1. Overview

With the boom of broadcast radio, TV channels and Internet, large volumes of audio or spoken documents are created and archived every day. Because of the difficulties and complexities of accessing and analyzing audio documents manually compared to text document, there is a growing need of using automatic audio processing technologies to efficiently index, search, access and analyze the information from audio data. The development of audio streaming also demands the real-time application of these technologies.

In many scenarios in presence of multiple speakers including conversations, meetings, conference and broadcast news, there are multiple audio sources or multiple speakers speaking within one audio file. Speaker diarization is the process used in these cases to segment an input audio signal into speaker-homogeneous segments, such that each speech segments are labelled by the corresponding speakers, as shown in Figure 1. Therefore, speaker diarization problem is often summarized to answer "who spoke when" question [1].

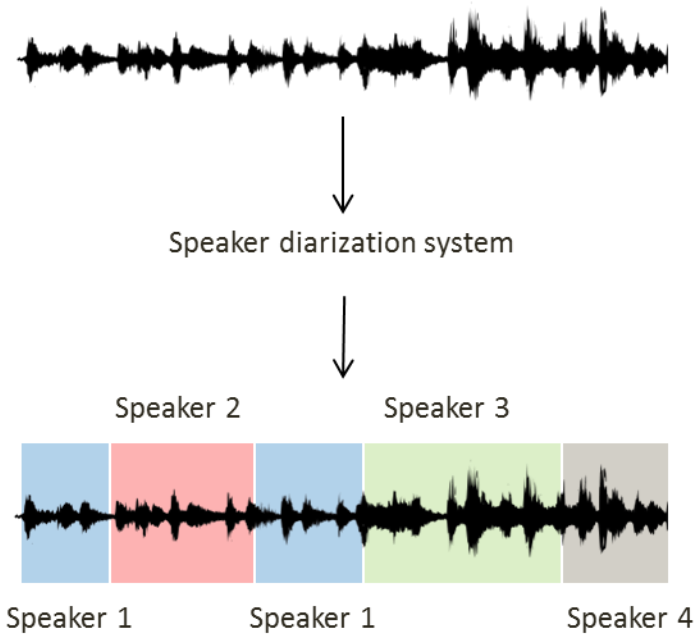

Figure 1 Overview of speaker diarization; the input audio signal is partitioned and sgements of same color belong to same speaker

The main difference between the speaker diarization and speaker recognition or speaker verification is that there is no prior speaker enrollment in the speaker diarization so speaker identities are completely unknown, therefore, the labels used for each unique speaker in speaker diarization are generic (e.g., Speaker 1, Speaker 2 etc., as shown in Figure 1). Another difference is that the temporal information is more important for speaker diarization than other speaker processing tasks.

Speaker diarization is a vital area in the community of speech processing because it provide the metadata in the audio of multiple speakers including the speaker segment

labels, position of speaker turns and number of speakers, which can provide more context of the speech and be used for information retrieval. In the scenario of two speakers, for instance, doctors and patients in medical recording or customer and customer service in telephone conversation, speaker diarization can be used as source separation tool so that the analysis or retrieval on the speech of each side can be performed more easily. Another important application of speaker diarization is to find the boundary of sentences in conversation and the corresponding speakers of the sentences, to improve the readability and overall accuracy of the automatic speaker recognition (ASR) system. Generally, speaker diarization is an important front-end tool such that the audio information output can be more efficiently used as input in multiple speech processing tasks including spoken document indexing and retrieval, speaker recognition and speech-to-text transcription [2].

The traditional three primary application domains of speaker diarization is telephone conversation, broadcast news and conference meetings [3]. The audio streams from these domains are different in style of the speech, style of the noise source, numbers and locations of microphones, configuration of environment and therefore present unique challenges. [4] makes detailed comparison between broadcast news and conference meetings. The majority of the literature in speaker diarization will only focus on one of the three cases and some propose specific techniques to tackle some unique problems. For instance, [5] propose the acoustic beamforming technology to take advantage of the multiple microphones available in the meeting room domain to facilitate the speaker diarization process. Therefore, the speaker diarization system that has advantageous

performance in one domain may not have comparable performance in other domains, and this domain-specific problem negatively affects the usability and extensibility of some systems.

Speaker diarization can also be referred as speaker segmentation and clustering, as the majority of diarization approach consist two main tasks of segmentation and clustering [6]. However, there is a recent work that excludes the clustering to perform fully supervised diarization [7]. The details of these two tasks will be presented further in the following sections.

1.2. Offline, Online and Real-time Diarization

While most of the past works aim at improving the accuracy of the speaker diarization system on recorded audio, there is limited work aims at improving the speed of the diarization system and the develop the speaker diarization system for real-time application.

Speaker diarization system can be differentiated as offline and online system. The offline system can access the whole audio recording before processing, and the clustering step is performed only when complete audio stream has been segmented. This means it is hard to adopt an offline diarization technique in real-time applications where the audio processing is required to be conducted simultaneously or with acceptable latency when the audio is input.

Online diarization, on the other hand, only have access to the audio data up to the point that is been recorded, which means the diarization have to perform in a "left-to-right" fashion [8] that process and assign the segments once they are created and detected in the audio stream. Therefore, online speaker diarization is more suitable for real-time applications. However, offline speaker diarization is still the focus in the field of speaker diarization [4] and there is limited work on online speaker diarization. A real-time speaker diarization system for the meeting environment is proposed in [9]. However, the system relies on the speaker seat locations and has the limitation of detecting only one speaker in one frame even if there are multiple speakers speaking. The online speaker diarization based on Gaussian mixture models (GMMs) and male, female and noise models, is tested with broadcast news data [10]. However, this system has difficulties dealing with speech overlapped by music. The novel Maximum a Posteriori (MAP) adapted transform within the i-vector speaker diarization framework proposed in [8], have a preferable diarization result for two-person telephone conversation audio, but still have diarization error rate (DER) 50% worse than offline system.

Generally, the performance of the online system is much worse than that of the offline system, but a state-of-art online algorithm proposed recently called Interleaved-State Recurrent Neural Network (UIS-RNN) algorithm is demonstrated to have performance that is comparable with offline diarization [7]. The UIS-RNN model that used to predict speakers' labels, is learned in a supervised manner, given the speaker label with temporal information and speaker-discriminative embedding (d-vectors) extracted from training data. Consequently, this process requires large training data that incorporate temporal data and the

diarization results will be significantly affected by the quality of the speaker embedding and not robust for different domains if the training data of UIS-RNN is domain-specific. Another obstacle for this method in real-time application includes the complexity and large computational resources required for the system.

1.3. Objectives

Given the limited research on the speed of the speaker diarization system and the gap of the real-time speaker diarization system, the general aim of this dissertation is to build a fast offline speaker diarization system that can label the speakers in recorded audio, and develop an advanced version of the system that can perform diarization in the real-time context.

More specifically, the proposed speaker recognizer system in this dissertation should fulfill the following requirements:

1) The offline system can perform speaker diarization for recorded audios like radio talk or phone conversation;
2) The real-time system can perform speaker diarization in a real-time fashion that can process the live speech audio and generate output as the input is analyzed. The system can correct the earlier output when necessary. Reasonable buffers of data or short delays are acceptable;

3) The system should be language-independent and operating-system independent; It is preferable if the system is domain-robust;
4) The system should not require the number of the speakers, identity of the speakers or voice samples of the speakers for the training;
5) The diarization results should be presented clearly in text form and being visualized. The audio playback should be available for result examination.

1.4. Scope of the Dissertation

As presented in Section 1.1, different application domains present unique diarization challenges. This dissertation will not aim at solving any domain specific challenges and will aim at building a system that is domain-robust for different domains. Broadcast news audio will be firstly considered for training or testing purpose in this dissertation, because of the availability of the audio data and the higher difficulties to analyzed, due to the fact that there are usually more people speaking comparing with the telephone conversation audio and there are more noise and interruption comparing with the conference meeting audio.

The scope of this dissertation is restricted to the audio signal processing techniques in speaker diarization context, so no information other than input audio signal itself, can be used in the proposed system. The speaker diarization techniques that incorporate information that is not from the audio signal such as environment configuration including seat and microphone location [9], and visual activity sensing including face tracking [11]

and focus of attention tracking [12], to assist speaker diarization, are all out of the scope of this dissertation.

Speaker diarization assumes no prior information of speaker identities and number of speakers for the input audio available. In this dissertation, this information will also be unknown and no speaker enrollment process should be designed in the system. The speaker recognition or speaker detection tasks that have the access to the voiceprint of the speakers within the audio, is out of the scope of this dissertation.

This dissertation will only consider the design of the software system without special requirement on the hardware architecture. The specialization framework proposed in [13] to perform parallel implementation of GMM training on GPU to speed up the diarization system falls outside the scope of this dissertation.

The scope of this dissertation will only focus on an independent speaker diarization system, without considering the combination of the speaker diarization system with other speech or audio processing system. The online diarization system that incorporates ASR decoder in the system design proposed in [14] is outside of the scope.

Because speaker diarization system commonly consists of multiple components, there are many literatures that aim at improving the performance of one specific component in the system, for instance, speaker activity detection (SAD), speaker modelling or clustering. However, in this dissertation, the emphasis will be put on the overall system design for

both offline and real-time context, and the investigation on the enhancement of the existing algorithm of specific components is subordinate.

1.5. Organization of the Dissertation

The remaining sections of this dissertation will be organized as follows.

Section 2 analyses the research problem, provide an overview of existing speaker diarization system and introduce the common components in the diarization system. The method to evaluate the speaker diarization system is also presented. Section 3 provides the design idea and adopted algorithm of the proposed offline and real-time speaker diarization system. The design of the visualization panel will be also presented in Section 3. Section 4 outlines the data, parameter, system configuration and the results of the experiment on our system. Section 5 concludes the dissertation with the discussion and summary of contribution and suggests the future direction of research.

2. Analysis of Problem

The aim of this section is to give an introduction of the existing speaker diarization problem. In Section 2.1, the generic speaker diarization system will be outlined. From Section 2.2 to Section 2.7, the role and functions of each common components in a speaker diarization system will be explained, the problem involved in our system design and possible solutions will be discussed. In the Section 2.8, the evaluation of speaker diarization system will be presented.

2.1. Generic Speaker Diarization Architecture

There are several previous works provide a comprehensive review on the existing diarization techniques and systems [2, 3, 15, 16]. The generic and prototypical speaker diarization architecture and its components that summarized by these works are illustrated form Figure 2 to Figure 4.

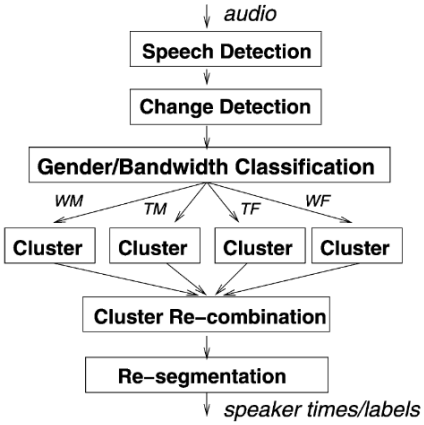

Figure 2 Generic speaker diarization architecture[1]

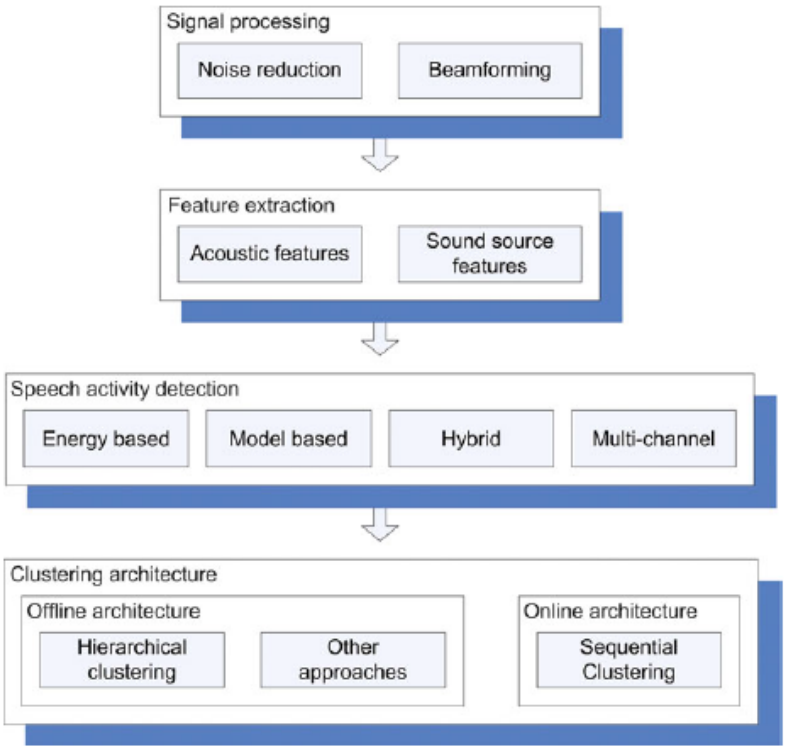

Figure 3 Generic speaker diarization architecture[2]

[1] The figure is extracted from [3]
[2] The figure is extracted from [15]

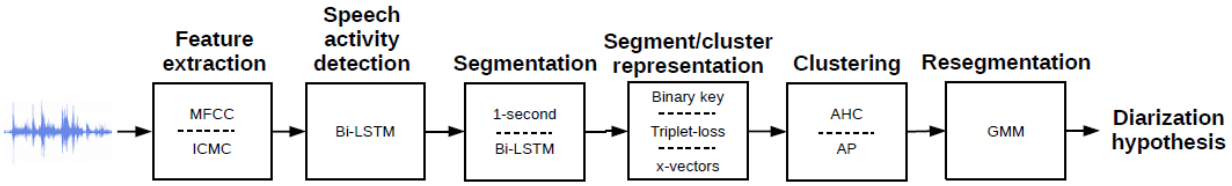

Figure 4 Generic speaker diarization architecture[3]

The common components of speaker diarization system in the literature consists of feature extraction, speech activity detection (SAD) or voice activity detection (VAD), segmentation or speaker change detection and clustering. Other components that may be incorporated in the system design includes domain-specific signal preprocessing such as noise reduction using Wiener filtering [17] and multi-channel acoustic beamforming firstly proposed in [18], gender/bandwidth classification [3], speaker embedding extraction or segment/cluster representation and post-processing such as resegmentation and SAD post-filtering [19].

The function of each component that exists in previous work and our system will be further in explained in Section 2.2 to Section 2.7 in the following.

[3] The figure is extracted from [16]

2.2. Voice Activity Detection and Speech Activity Detection

Voice activity detection (VAD) or speech activity detection (SAD) is commonly used as the preprocessing step in various speech processing tasks including speech recognition, speech verification and therefore is included in the majority of the speaker diarization system. It is used to identify the region of the audio data that is being voiced or containing speech, from the unvoiced or non-speech regions that contain silence or background noise.

Energy-based voice detection and model-based voice detection are two main approaches of VAD. The energy-based voice detection removes the silence part based on the energy level and has the advantage of simplicity and speed. However, this approach fails to distinguish the load noise from the speech, and therefore is ineffective in many application domains of speaker diarization [20, 21]. To avoid the limitation of energy-based detection, model-based detection that is developed on the different acoustic phenomena, are more frequently used in speaker diarization system [15]. The dominant energy-based approach is the use of GMM. The simple SAD approaches divide the input signal into speech or non-speech regions using the GMM to model two different classes while the more complex approaches will use GMMs to model more voice classes including gender (male/female), bandwidth conditions, music and noise. One example is the SAD process to use five models for different type of non-speech voice and three models for speech [22].

As explained in Section 1.2, the attempt to improve current SAD/VAD accuracy is out of the scope of this dissertation. Therefore, we will directly implement the suitable approach in the system. Model-based approach will be used in the system design given the limitation of energy-based approach. The speed and simplicity of implementation is the top factors to consider choosing suitable algorithm.

2.3. Acoustic Feature Extraction

The raw audio data is usually converted into a sequence of acoustic feature vectors that contains speaker specific information before the segmentation and clustering. This feature extraction step try to acquire the acoustic features that contain formant information, model the mode of excitation and the shape of the vocal tract when people producing speech [15].

The cepstral-domain features that are generated from short-time segment (typically 10-30ms) of the speech is the most popular techniques in the community of speaker diarization. These features are capable of capturing the energy fluctuation in different frequency bands and resonance properties of vocal tract [15]. Common features include Mel Frequency Cepstral Coefficients (MFCC) [23], Perceptual Linear Predictive Coefficients (PLPC) [24], Linear Prediction Cepstral Coefficients (LPCC) [25] and constant Q transform Mel-frequency Cepstral Coefficients (ICMC) [26] . These features are different in frequency analysis and frequency smoothing techniques and the

comparison and evaluation of these acoustic features in speaker identification task is presented in [27].

Among all aforementioned features, one of the most commonly used features in speech processing task is MFCC, which is firstly proposed in [23]. The algorithm to extract MFCC from the audio signal is briefly introduced as below.

Firstly, the audio signal is divided into short windows and the Fast Fourier Transform (FFT) is used for each window to transform the signal in time domain to frequency domain.

Secondly, the spectrum of each window after the FFT process is filtered by Mel scale filters banks. The Mel scale frequency is proposed to reflect how the humans perceive the frequency in speech and this scale is approximately linear below 1 kHz and logarithmic above 1 kHz [28]. The given frequency f in linear scale and measured in Hertz (Hz) is therefore converted to frequency in Mel scale $Mel(f)$ by the approximate formula designed as below:

$$Mel(f) \approx 2595\, log\left(1 + \frac{f}{700}\right) \quad (2.1)$$

Finally, Discrete Cosine Transform (DCT) will be used to convert the logarithmic Mel spectrum obtained from last step to the Mel frequency cepstral coefficients (MFCC) in time domain [29]. The corresponding formula to calculate final coefficients C_n can be presented as,

$$C_n = \sum_{k=1}^{K} (log\, S_k)\, cos[n(k - 0.5)\frac{\pi}{K}] \quad (2.2)$$

, where K is the total number of filters in the Mel filter bank and S_k is the output power of the k^{th} filter.

2.4. Segmentation

Speaker segmentation aims at splitting the original audio stream into segments containing one active speaking speaker and many other literatures adopt speaker change detection (SCD) techniques to fine the speaker change points in the audio [2, 3] . The popular approach to segmentation tasks is to firstly calculate a distance metric between the adjacent windows of audio data and secondly decide whether there is a speaker change based on the distance. These approaches are categorized as metric-based segmentation [15].

One popular distance metric used in segmentation is Bayesian information criterion (BIC) firstly introduced in [30] and firstly used in speaker segmentation in [31]. Many state-of-art systems incorporate BIC as segmentation metric [18, 32]. As BIC approach is computationally intensive, several works (e.g. [33]) propose modification or other technologies used with BIC to speed up the process. Some common alternative distance metrics include Generalized Likelihood Ratio (GLR) [34] and Kullback–Leibler divergence [35]. Some recent papers propose advanced machine learning technology including deep neural network (DNN) to find speaker change points [36].

2.5. Speaker Modelling

The speaker modelling is the heart of the many state-of-art speaker diarization system, and the similar step is called embedding extraction in some other research. This step aims to find speaker-discriminative identifiers that can be used to distinguish unique speaker from each other. Some models or embeddings are first found to be effective in speaker recognition or identification tasks, and then are adopted in speaker diarization tasks. Famous speaker models used in state-of-art system include Gaussian Mixture Model (GMM) and Hidden Markov Model (HMM) and famous speaker embeddings include speaker factors [37], i-vectors [8] and d-vectors [38, 39].

2.6. Clustering

Clustering is another important component in speaker diarization focusing on agglomeration of segments from segmentation step into groups that from the same speaker. Clustering approaches can be divided into two categories: offline clustering and online clustering, which determine whether the system can be performed in offline or online manner.

One of the most common offline clustering approaches in the literature of speaker diarization is the bottom-up approach of agglomerative hierarchical clustering (AHC) [2]. This approach starts at certain number of clusters and successively merges the clusters and reduces the number by one at each iteration until only one cluster is left. Assume the

initial number of cluster is K, then the iterative process generate a set of clustering solutions $\mathrm{C} = (C_1, C_2, \ldots, C_K)$ with decreasing number of clusters, where solution C_1 has K clusters and C_K has one cluster. Then some clustering selection technique is used to select the best clustering solution from C.

Another popular approach proposed recently is unsupervised i-vector clustering. [40] propose a system that uses i-vectors and probabilistic linear discriminant analysis (PLDA) which has preferable performance for multi-language telephone conversation data.

Online Speaker Clustering

The online speaker clustering is different from the offline clustering that not all segments in the audio recording are available and new segment need to be labelled by the clustering process when it is created. This difference leads to two distinct components in online speaker clustering: novelty detection component that detect the new speakers and incremental learning component that update the speaker models for new data or observation [15]. The online speaker clustering techniques can achieve linear complexity to the number of audio segments therefore it is more time-efficient than the offline hierarchical speaker clustering [41].

2.7. Resegmentation

The resegmentation is found as the last step in many speaker diarization systems, which is used to refine the segment boundaries and clustering results from the previous steps, especially when the uniform segmentation is performed. The common approach of resegmentation is to use Viteribi algorithm with MFCCs [42]. [43] and [44] shows resegmentation with GMM Viterbi helps in reducing diarization error rate. [45] analyses a new resegmentation approach using factor analysis subspace and demonstrate slightly better performance compared with tradition Viterbi approach.

2.8. Evaluation of Speaker Diarization

Diarization error rate (DER), defined by U.S. National Institute of Standards and Technology (NIST) [46], is the primary performance evaluation metrics commonly used in NIST Rich Transcription (RT) diarization tasks and the community of speaker diarization research. Given the system speaker labels generated a speaker diarization system and the reference speaker labels generated by the ground truth, the optimal mapping of reference labels to system labels is found and the DER is then calculated as the ratio of incorrectly mapped speech time, which consists of three following components:

- Missed speaker time T_{miss} ,which is the time when more reference speaker labels than system speaker labels

- False alarm speaker time T_{FA}, which is the time when more system speaker labels than reference speaker labels
- Speaker error time $T_{spk-error}$, which is the time when system speaker label are not matched reference speaker labels

Given the total speech time of the audio (T_{speech}), the DER can be considered as:

$$DER = \frac{T_{miss}+T_{FA}+T_{spk-error}}{T_{speech}} \quad (2.3)$$

A more formal formula to calculate DER defined in [46] is as below:

$$DER = \frac{\sum_{all\ segs}\{dur(seg)\cdot(max(N_{ref}(seg),N_{sys}(seg))-N_{correct}(seg))\}}{\sum_{all\ segs}\{dur(seg)\cdot N_{ref}(seg)\}} \quad (2.4)$$

, where for each segment seg:

$dur(seg) =$ the duration of seg,

$N_{ref}(seg) =$ the number of reference speaker labels in seg ,

$N_{sys}(seg) =$ the number of system speaker labels in seg ,

$N_{correct}(seg) =$ the number of reference speaker labels in seg that their mapped system speaker labels are also in seg.

3. Design and Construction of Software System

In this section, the discussion of the design and construction of the proposed software system will be divided into three subsections: the design of the offline version of the speaker diarization system in Section 3.1, the design of the real-time speaker diarization system in Section 3.2 and the design of the visualization panel in Section 3.3. The algorithm that is applied and the reason to choose a certain algorithm would be discussed in each section. The programming language and the third-party package used in the design of the system are presented in the Section 3.4.

3.1. Design of the Offline Speaker Diarization System

3.1.1. Overview

The pipeline system used in the design of the proposed offline system is the diarization system incorporating Binary Key (BK) speaker modelling techniques, firstly proposed in [47] and later upgraded in [48]. The system overview is illustrated in Figure 5.

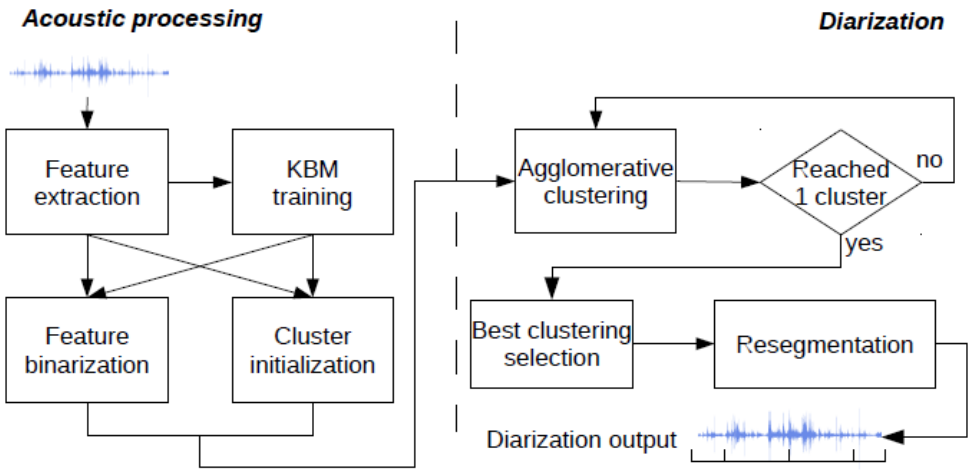

Figure 5 Overview of diarization system pipeline[4]

The pipeline system is an offline speaker diarization system. Suitable modifications and upgrade on the pipeline system will be made for our proposed system. In this section, the workflow and each components of the proposed offline system will be presented. The proposed real-time system will be designed based on the offline system we proposed, and the details will be discussed in the Section 3.2.

The workflow of the proposed offline speaker diarization system is presented in Figure 6 and it consists of below main blocks:

[4] The figure is extracted from [48]

1) Voice activity detection block to detect and remove the regions of silence from the input audio
2) Acoustic feature extraction block to compute the MFCCs from non-silent region of the input data to form to a features vector
3) Segmentation block to divide the non-silent audio regions into segments that contains one active speaker
4) Binary Key Background Model (KBM) Training block to train the MFCCs from step 2) to obtain the KBM that model the acoustic space of the audio
5) Feature Binarization block to transform the vector of features of segments or clusters into Cumulative Vector (CV) and Binary Key (BK), which is used as the representation of segments or clusters
6) Clustering block to merge the segments into clusters of homogenous speakers based on the CVs or BKs
7) Resegmentation block to refine the clustering result

Step 4) and 5) constitutes the BK speaker modelling, which is the heart of this system.

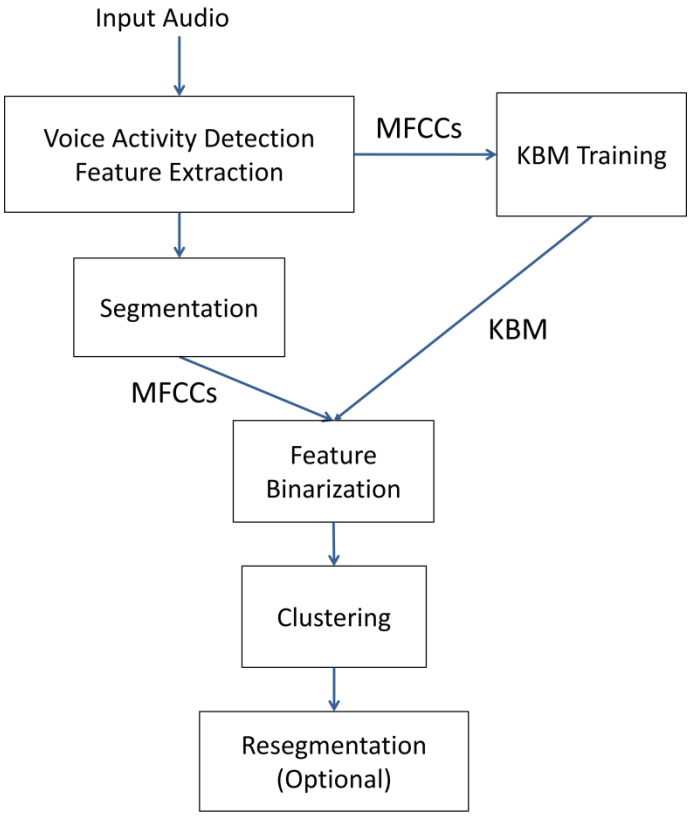

Figure 6 Offline system workflow

The role of each step and the classic design in existing system will be presented in the following. The algorithm we have adopted in our system and the corresponding design idea will also be further explained respectively.

3.1.2. Voice Activity Detection

The model-based VAD algorithm that Google developed for the WebRTC project [49] will be applied in the design of our system, the reason includes:

1. The WebRTC project targets on real-time capabilities and its VAD algorithm has been widely used for different delay-sensitive scenarios [50], which is suitable for our offline and real-time applications.
2. The free and open-source implementation of the algorithm is available and there is Python interface (Py-webrtcvad [51]) that is well-suited for our development.

The VAD process in the system can be performed on the all audio data for the audio recording or the new input data for real-time audio stream. The result of the VAD is a mask vector that used to exclude the data segments or the acoustic features that contains no speech from the original data.

3.1.3. Feature Extraction

In this dissertation project, the focus will be put on the MFCCs which are frequently used in the community of speaker diarization and more specifically, the online speaker diarization system [10, 8]. Librosa, the python library for music and audio analysis [52, 53], will be used to extract the MFCCs. The python code is simple and straightforward as below:

```
Features = librosa.feature.mfcc(y = input_data, sr = sampling_rate,
dct_type=2,n_mfcc = ncoeff, n_mels = nfilters, n_fft = length_of_fft_window,
hop_length = hop).T
```

The parameters used for MFCCs extraction will be discussed in Section 4.1.3.

3.1.4. Segmentation

Although many systems use various distance metric to decide whether the speech of the adjacent windows are from the same or different speakers, a simple method that use fixed-sized windows to splits the audio into small equal-sized segments, will be used in our system.

3.1.5. BK Speaker Modelling

In this project, we will adopt the Binary Key (BK) speaker modelling techniques that is firstly proposed in [54] and later used in speaker diarization context in [47]. This technique has advantages of being domain-robust, requiring no external training data and running faster than real-time, so therefore can be utilized for our offline and real-time system.

The process of BK speaker modelling can be divided into two steps: KBM training and feature binarization, which are further explained as below.

KBM Training

The first step of the binary key speaker modelling is to train a GMM-like model called KBM from the acoustic features (MFCCs for our system) extracted from the input data. While the majority of the speaker modelling or speaker embedding extraction techniques

requires large amount of external training data that is from the same domain of the tested data, the BK techniques train the KBM directly from the input tested data. Therefore, BK modelling is domain-robust and can avoid the negative impacts from the mismatch of the acoustic conditions between training and tested datasets.

The process of KBM training is illustrated in Figure 7. To obtain the KBM, multiple Gaussians are trained on the data that is separated by a 2-second sliding window on the input features data. The mean and the standard deviation of the 2-second data will be calculated to get the Gaussian, which will be realized by *multivariate_normal* function from Scipy library [55] in Python. The shifting rate of the sliding window is determined automatically to have enough number of Gaussians. All resulted Gaussians form a Gaussian pool that covers all acoustic space in the input audio. Then *N* Gaussians are selected from the pool using single-linkage clustering strategy with cosine similarity as the similarity measures. The first Gaussian is selected with maximum likelihood and the Gaussians with highest dissimilarity with previously selected Gaussians is selected subsequent until getting *N* Gaussians. Then the KBM is formed by these *N* Gaussians, which are considered to be the most complementary and discriminat Gaussians that can be used to represent the speaker acoustic space. The number of Gaussians *N* can be a fixed number or be determined relatively by a percentage of the total number of Gaussians in the pool.

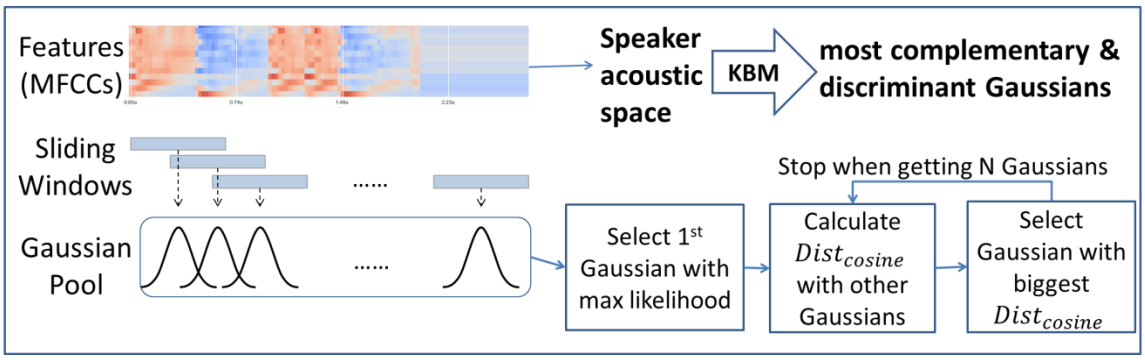

Figure 7 KBM Training

Although KBM is a GMM-liked model and the GMM that is trained by some external data can also be used to produce speaker discriminative representation, [47] and [54] demonstrates that the KBM outperform classic GMM in distinguishing the speakers. Moreover, KBM has the advantage of lower computational cost and consequently shorter processing time, comparing the expectation-maximization algorithm in GMM training.

Feature Binarization

The second step of the binary key speaker modelling is the feature binarization that transforms feature vectors of an utterance to into a binary key, and this process is shown in Figure 8.

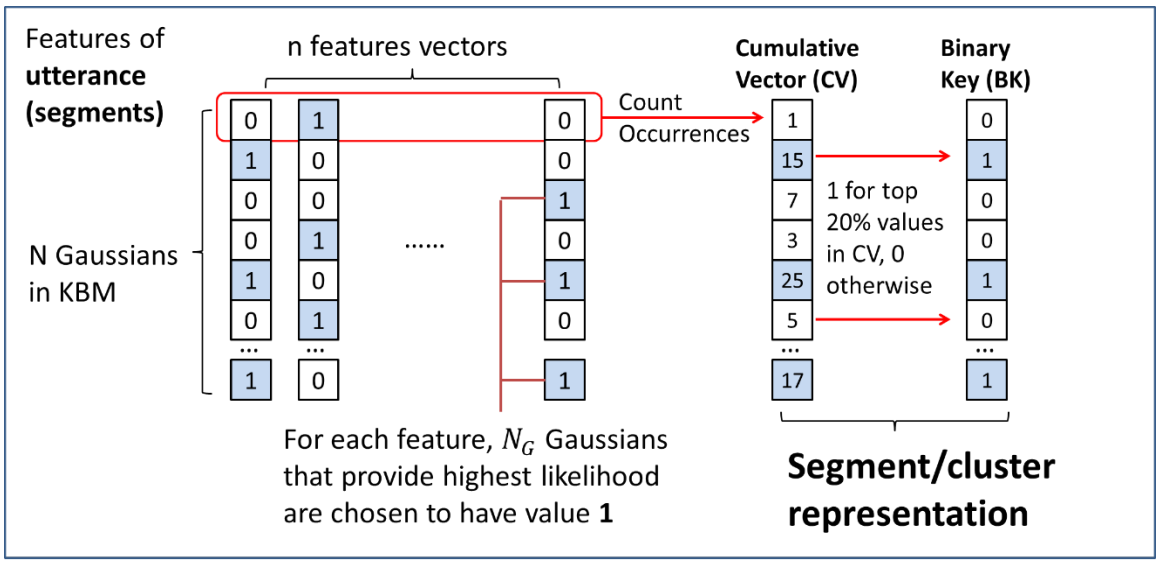

Figure 8 Feature Binarization

Firstly, a matrix with number of rows equal to *N* (size of KBM) and number of columns equal to number of feature vectors are created. For each column of each feature vector, N_G rows representing the Gaussians that provide highest likelihood for the given features are chosen to have value 1. All other rows in this column are set to have value 0. This process recorded the best N_G Gaussians from the KBM that represent each feature. Secondly, we sum the values of each row, to form a vector called Cumulative Vector (CV) of size *N*. This CV contains the count of the occurrences of each Gaussians has been selected as the top-likelihood Gaussian for all feature vectors, so intuitively the larger number in the CV indicates higher impact of the corresponding Gaussian components. Thirdly, the binary key (BK) is obtained by setting the value as 1 for the positions that have top M% values in CV and the value as 0 for the other positions. For example if M

equals to 20, then the positions whose values in CV are over 80^{th} percentile will be set to have value 1 in BK.

The resulted CV or BK can be used to represent the input utterance and theoretically, the utterances from same speaker will have similar Gaussians in the KBM that have highest impact in modelling the speech, and therefore generate similar CVs or BKs. The input series of features in this step can come from the utterance of a short segment or a speaker cluster. Segment assignments and clustering can be performed by comparing the CVs or BKs from the segment or clusters by some similarity measures.

While the CV store the relative importance of the Gaussian components in KBM, the BK only store the Gaussians components that have greatest impact to model the input features. Therefore, there is information that is missing in the process of transforming the CVs into the BKs. In this project, CV will be considered over BK, as [56] found that CV is more speakers discriminative as the segment / cluster representation comparing with BK.

Because the value in the CV is the number of occurrences of the corresponding Gaussian has been selected for the input feature vectors, the longer feature vectors generally results in CV of larger magnitude. Consequently, to compare two CVs from feature vectors of different sizes, the angle instead of the magnitude of the vectors should be considered. The cosine similarity is therefore proposed as the similarity measures to compare CVs. The computation of cosine similarity between two vectors of positive integer values is simple and fast and the formula is:

$$S_{cos}(x, y) = \frac{x \cdot y}{\|x\|\|y\|} \qquad (3.1)$$

$$, where\ S_{cos}(x, y) \in (0, 1)$$

The value of cosine similarity is between 0 and 1, where values close to 1 indicate high similarity between two CVs, while values close to 0 indicate high dissimilarity.

In programming levels, cosine similarity can be easily computed by cdist function from Scipy library [55] and the pseudo code is:

```
from scipy.spatial.distance import cdist
S = 1 - cdist(CV1, CV2, metrice = 'cosine')
# CV1, CV2 is ndarray
```

[48] has demonstrated the system using cosine similarity have more preferable performance comparing system using KL2 divergences between Gaussians. Moreover, the cosine similarity for the CVs has the advantages of simplicity and speed.

Because only the CV will be used to represent segments or clusters in our proposed system and BKs will be ignored, therefore this feature binarization step will be called as representation / CV calculation in the following section.

3.1.6. Clustering

The offline clustering adopted in this system is the AHC mentioned in Section 2.6. The workflow of the AHC in my system is illustrated in Figure 9.

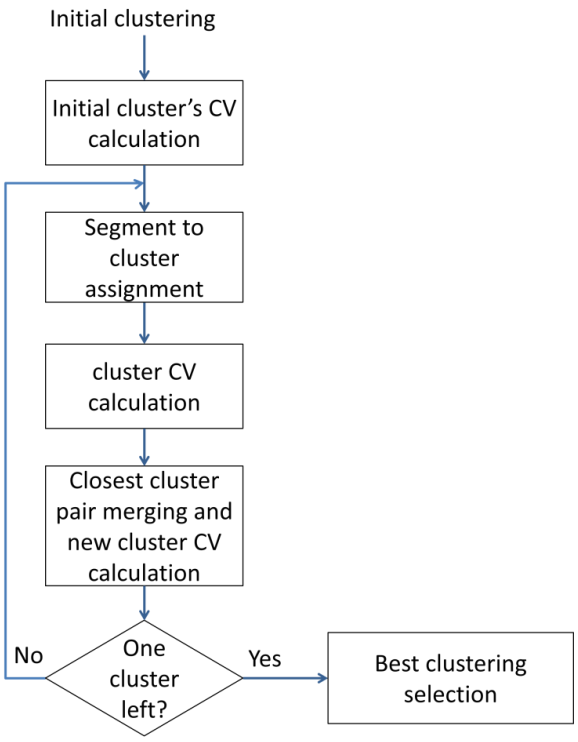

Figure 9 Agglomerative hierarchical clustering

There are many cluster initialization techniques have been studied in previous studies and one of the most common and simple approach is to divide the input audio data into a few equal-sized chunks. This uniform initialization method generally results in equivalent performance found in [57] and has advantages of simplicity and speed. The initial number of clusters K should be larger than K_{opt} , the optimum number of clusters for the audio.

After the cluster initialization, the CVs for the initial clusters are calculated as the cluster embedding using the techniques in Feature Binarization. Then the audio segments are

reassigned to clusters, by comparing the CVs of the segments to the CVs of the current cluster by cosine similarity. The segment is assigned to the cluster if highest cosine similarity between their CVs is achieved. After the reassignment, CVs are calculated for the new clusters. Then the cosine similarities will be calculated between the CVs of the new clusters and the cluster pair with highest similarity is merged. The total number of clusters is consequently reduced by one. The CV of the new clusters after merging will be calculated. If there is more than one cluster is left, the iterative process will continue from segment to cluster assignment. For each iteration, the clustering results will be stored to form a set of clustering solutions with size K.

Because the segment size is fixed and equal, the calculation of CVs for each segment only needs to be performed once. CV of segments can be used to obtain the CV of cluster. Therefore, the calculation of CV in this clustering step requires low computational cost. The main computational cost of this step is put on the calculation of cosine similarity between segment and cluster, or between cluster and cluster. The overall processing time required for clustering is short and it is demonstrated in Section 4.3.1.

The best clustering block in Figure 9 is responsible for selecting the best clustering solutions from all clustering solutions generated from previous iteration. Classic criteria to select best clustering include Generalized Likelihood Ratio metrics [34], Kullback-Leibler based metrics [58], and T-test T_s metric [59]. In this system, the elbow criterion will be applied to choose best number of clusters by evaluating the curve of within-class sum-of-squares (WCCS) for each number of clusters.

3.1.7. Resegmentation

As the fixed-sized uniform segmentation is performed, the boundaries of the segments produced may not be precise, resegmentation block to refine the boundaries is useful for the offline system. The traditional practice to use GMM for resegmentation will be adopted. The GMM are trained to model the clusters from the clustering steps. The frame of MFCC sequence will be evaluated with GMMs and then the frame will be assigned to the cluster when the maximum log-likelihood is achieved.

3.2. Design of the Real-time Speaker Diarization System

The proposed system in the Section 3.1 is an offline system and several modification and adaptions are required to make the system to process the audio data in the real-time context. In the Section 3.2.1, the discussion on each component in the offline system will be made to see whether these components should be kept, modified or excluded for the real-time system. In the Section 3.2.2, the workflow and the details of the proposed real-time system are presented.

3.2.1. Discussion of Components in Real-time System

For the VAD and feature extraction, the process can be performed in the online manner (i.e. the strict left-to-right manner). Both VAD and MFCC extraction can be performed on the whole recorded audio or on the newly input data. Therefore, this process can be

performed whenever new data is inputted and no special modification needs to be made for the real-time system.

For the KBM training, the process is performed on the all data of the recorded audio to model the whole acoustic space of the input audio in our offline system. In the real-time situation, KBM can also be performed on the newly input data combined with data stored before. However, two challenges exist. First, the minimum size of the initial Gaussian pool requires at least 10-15 seconds audio data for the KBM training. This means we need to have some data in the beginning of the real-time audio stream as the buffer to train the KBM and the diarization results are not available in the real-time for these data. This is acceptable as the time buffer required is short. In the application that a short buffer is needed, the minimum size of the Gaussian pools can be lowered. The second challenge of KBM training in real-time context is that the although the KBM training is fast, this process will still lead to delay of the results when data stream is long because training is processed on all data that is available. Since the KBM training is not necessary for every second of data in the real-time data stream, a possible solution to this challenge is to use a separate thread to train the KBM constantly so that consequent feature binarization and clustering can use the up-to-date KBM without being stuck by the KBM training.

For the features binarization, the computation of CVs can be performed on the features of all sizes so therefore can be performed in an online manner theoretically. However, as the KBM is updated in an offline manner and the corresponding CVs need to be recomputed

for all the features available whenever a new KBM is trained, feature binarization should be run in an offline manner. This re-computation of CVs for all features is time-consuming in real-time context, therefore, this step will be run using a separate thread together with KBM training, in the real-time system.

For the clustering, the AHC clustering method of the proposed system is an offline clustering method so modification is necessary. The proposed solution is to use a fast and naive online clustering method based on threshold first for the new data, then the offline clustering that run on another thread will be used to update the previous result when it complete. This method allows the results of the online diarization have the quality of offline clustering without causing too much delay.

For the resegmentation, this step can only be performed in an offline manner when all segments and clusters are available. Considering the speed constraints in real-time situation and the limited improvement of the performance, this step will be excluded in the proposed real-time system.

In summary, the real-time version of the system combines the components that operate in online manner including VAD, feature extraction and online clustering, with components that operates in offline manner including KBM training, feature binarization and AHC clustering. Consequently, both the speed requirement for real-time processing and the quality of the offline processing can be achieved.

3.2.2. The Workflow and Details of the Real-time System

Based on the discussion in the last section, the workflow of the proposed real-time speaker diarization is showed in Figure 10.

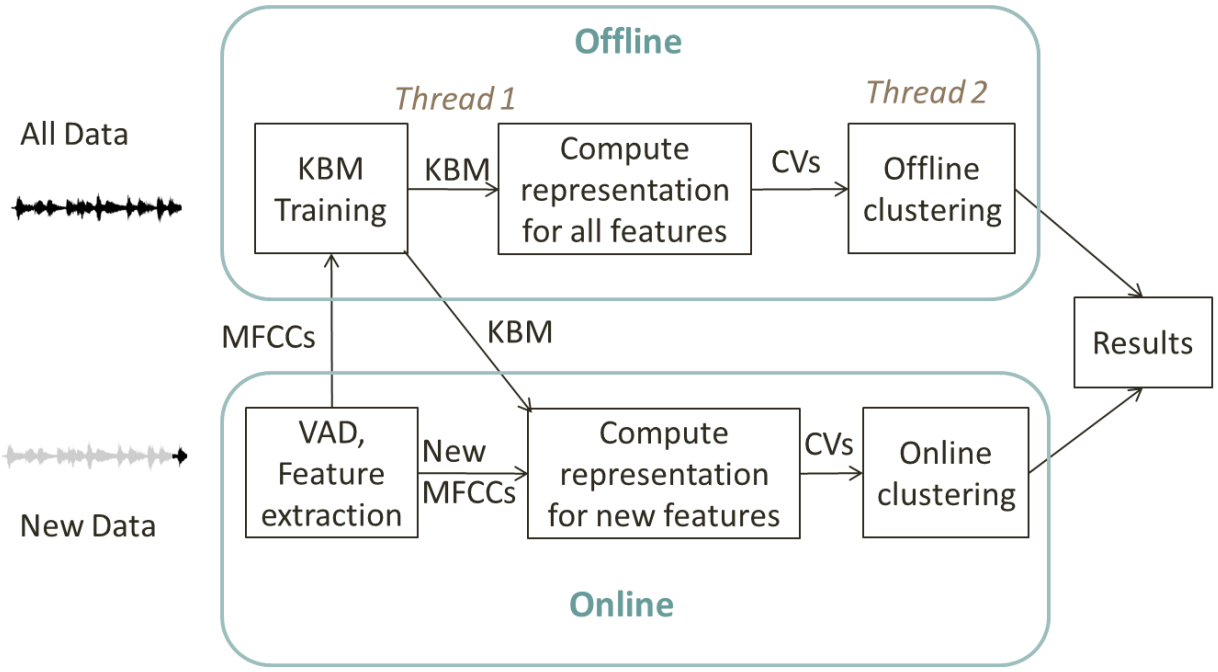

Figure 10 Real-time system workflow

Firstly, the new data inputted into the real-time system from the real-time data stream will go through the VAD and feature extraction process to obtain the MFCCs from voiced data. The MFCCs will be used for KBM training and feature binarization (representation computation) later.

Secondly, a separate thread (Thread 1) will perform KBM training on all available MFCCs. This KBM training process will start once the size of features vector is large

enough to generate required minimum sizes of Gaussians pool and will be repeated whenever there are new MFCCs available. After the new KBM is trained, the representations (CVs) will be computed for all available MFCCs, including the MFCCs extracted from the new data. The total number of CVs will be equal to the size of the MFCCs vectors extracted from all available data.

Thirdly, another separate thread (Thread 2) will perform the offline clustering (AHC) based on the CVs from the last step and repeat the process when new data is inputted. The online clustering will be used on the audio data that is not labelled by the results from the offline clustering. Therefore, the final diarization result will consists of both results from offline clustering and results from online clustering and labels all audio data we obtained from the real-time audio stream.

3.3. Design of the Visualization Panel

A visualization panel for the speaker diarization / recognizer system can help the users to understand and examine the diarization results intuitively. The target of the design of the visualization panel includes:

1. The number of the speakers is shown;
2. Different speakers and their speech on the timeline can be distinguished easily;
3. Spectrogram can be shown with the diarization results;

4. For the offline diarization system, the diarization results are shown once the diarization process, while for the real-time diarization system, the results can be shown and updated every 1s until the diarization process ends;
5. For the offline diarization system, the users can choose the start point at the timeline to playback the audio and pause it in anytime, while for the real-time diarization system, the tested audio can be played automatically when diarization process starts;
6. Same programming language as the speaker diarization system is used.

To complete the above targets of the visualization panel, two python modules viewer.py and player.py are designed respectively.

The player.py is the module to open, play and pause the input audio files. Python packages Pyaudio is incorporated in the script to realize the necessary functions of the audio player.

The viewer.py will be used to show diarization results of the input audio or audio stream. The x-axis is the timeline of the audio while the y-axis shows the number of the speakers. Rectangle of different colors will be used to display different speakers and their speech in the timeline. The position and the length of the rectangles will be determined by the position and the length of the speech in the timeline respectively. The Matplotlib library, which is the most popular library in Python for 2D plotting will be used in this script.

Different viewer.py script will be used for offline and real-time system because different needs of visualization for these two systems.

The demonstrations of the visualization panel for offline system are illustrated in Figure 11 and Figure 12. The red line in the graph indicates the current time point of the audio and will be keep moving as the audio is playing.

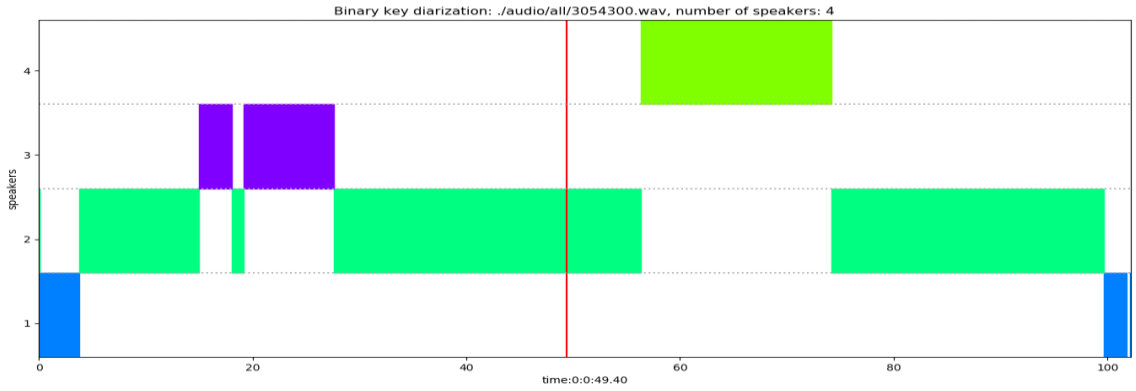

Figure 11 Visualization panel for offline system

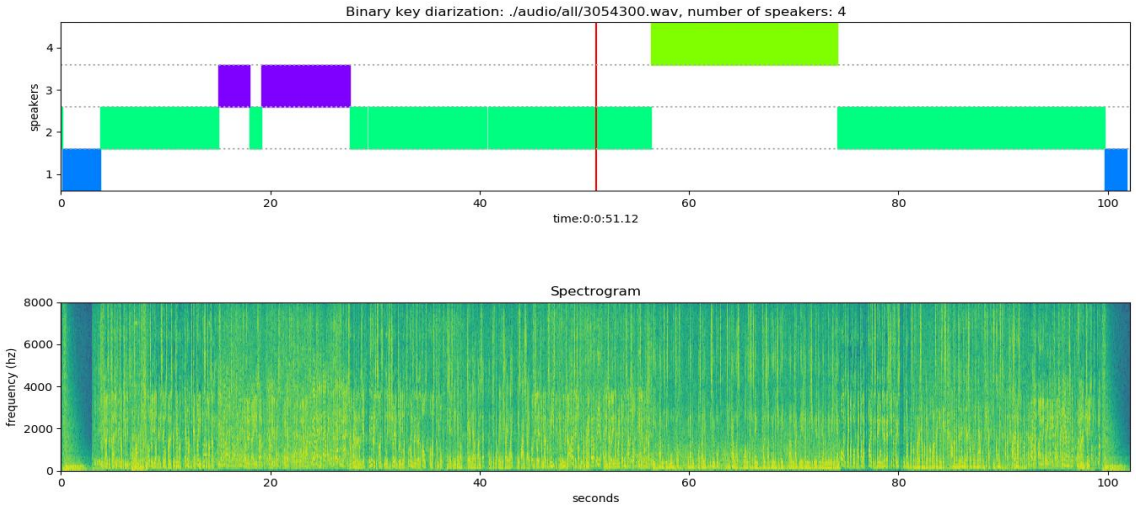

Figure 12 Visualization panel for offline system, with spectrogram shown

The demonstrations of the visualization panel for offline system are illustrated in Figure 13 and Figure 14. As the diarization will generate results within 1s for every 1s new data, the visualization panel will update the result every 1s as well. The rectangle block to shown diarization results and the length of the spectrogram will be increasing as more diarization results are available.

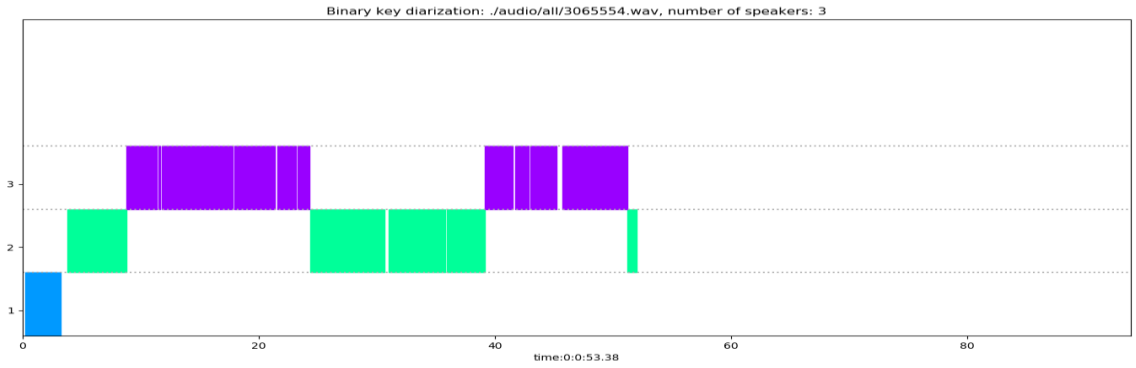

Figure 13 Visualization panel for real-time system

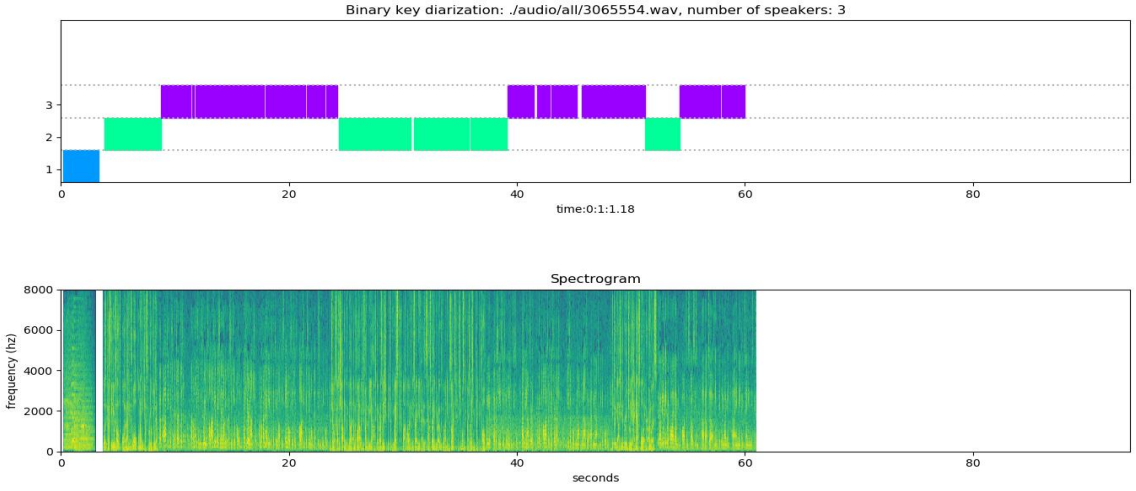

Figure 14 Visualization panel for real-time system, with spectrogram shown

3.4. Programming Language and Third-party Package

All code used to develop above diarization system is written in Python 3.7. The third party python package that used in the system includes:

- Numpy
- Scipy
- Librosa
- Py-webrtvad
- Pyaudio (only for audio playbacks)

All these packages are well tested in multiple operating systems including Windows and Linux. Although the proposed diarization system proposed is tested only on Windows, it is operating-system free that can be easily implemented on the other platform.

4. Experimental Results and Discussion

In this section, the experimental results of the offline version and real-time version of the proposed speaker diarization system will be presented. In Section 4.1, the setup of the experiments including testing environment (hardware configuration and operating system of the computer that conducts experiments), data, parameters and the methodology to simulate the real-time data stream using recorded audio will be introduced. The experimental results of the DER of the proposed offline and real-time system are presented in Section 4.2. Results and discussion on other experiments focusing the processing time of the system are presented in Section 4.3.

4.1. Experiment Setup

4.1.1. Experiment Environment

The experiments of the proposed speaker diarization system are conducted on a personal laptop with Intel® Core™ i5-4200U CPU @ 1.60GHz 2.30GHz and Windows 8.1 operating system. This laptop is purchased in 2013 and the system configuration is somewhat outdated. It can be argued that the faster running time can be achieved by the computer with higher processing and computational power.

4.1.2. Data

The data used for testing purpose includes the broadcast news audios from SAIVT-BNews database [60] and meeting audios from ICSI Meetings Recorder corpus [61]. All data are free for research purpose and used for testing or training in other speaker diarization research. All tested data have a sample rate 16 kHz.

4.1.3. Parameters Setting

The parameters used in the proposed systems and their values are summarized in Table 1. All values are set based on previous empirical studies.

Parameter	Value	Explanation or remark
Window length for feature extraction	25 ms	MFCC is the short-term spectral features that extracted from the short window (typically 20-30ms) of audio
Frame shift for feature extraction	10 ms	The length of the resulted MFCCs length = audio duration / 10ms the hop length (number of samples between successive frames when extracting MFCCs)

		= sampling rate * 0.01
Number of mel filters used	20	19-dimensional MFCCs using a 20-channel Mel-filterbank
Number of MFCCs employed	19	
Sliding Window length for computing Gaussians	0.02s	The shifting rate of the sliding windows on the MFCCs is determined based on the window length and audio duration to ensure there will be 1024 Gaussians trained
Minimum number of Gaussians in the initial Gaussian pool	1024	
KBM size	0.3 * Gaussian pool size	The resulted KBM size would be around 300 - 350, which is empirically the optimal KBM size that achieve lowest DER
Top Gaussians per features	5	Number of Gaussian components selected for top-likelihood foe each features in feature binarization step
Segment size; Segment increment after and before; Segment shifting	1s	When commutating the CVs for the input futures vectors, segments of 1s augmented 1s after and before (totaling 3s) will be used.
Number of initial clusters	16	The number of initial clusters should be larger than the maximum number of

		speakers in the tested data
Number of GMM components in resegmentation	6	These are parameters for resegmentation. Larger EM iterations and larger number of GMM components require more time to process
Number of expectation-maximization (EM) iterations	10	

Table 1 Summary of the parameters and their values

4.1.4. Simulation of Real-time Data Stream

In order to test the performance of the real-time system using multiple audios, it is favorable to simulate the real-time data stream using the recorded audio that is used in offline system experiment. Our approach to the simulation is proposed in this section and is illustrated in Figure 15.

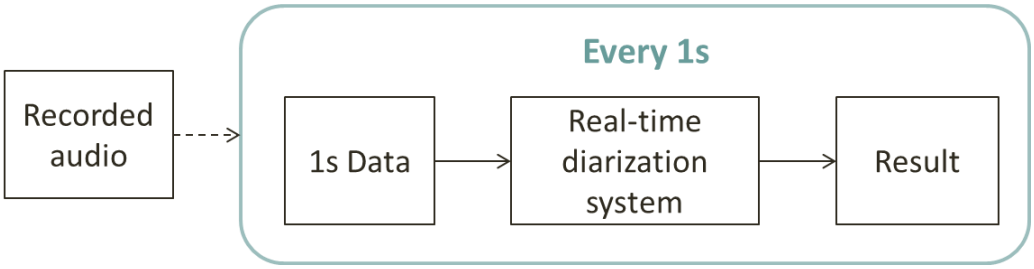

Figure 15 Simulation of real-time data stream using recorded audio

The data of the recorded audio will be fed into the real-time diarization system every 1s, and the system is required to generate the result within 1s. Since all tested audio has a sample rate of 16 kHz, 16000 data points will be fed into the system for every 1s. At time t, the real-time diarization system can only have the audio data up to time t and is required to generate the speaker labels up to time t. The total time for the system to process an audio file is the same as the duration of this audio file.

4.2. Experiment on Diarization Error Rate (DER)

In this Section, the experiment to examine the diarization error rate (DER) will be performed on the proposed offline and real-time system. The system will generate results into text files in Rich Transcription Time Marked (RTTM) format, which is commonly used in evaluation of the diarization system. The ground-true speaker labels and corresponding segments are found for the tested audios and parsed into RTTM format to have the reference result. A script names "md-eval-v21.pl"[5] developed using Perl language by NIST is used to evaluate the DER of the system's result compared with the reference's result.

The DER of the offline system and real-time system will be presented respectively in Section 4.2.1 and Section 4.2.2. In order to compare the system performance, we implemented one of the most advanced diarization algorithm UIS-RNN developed by

[5] md-eval-v21.pl is available at https://github.com/jitendrab/btp/blob/master/c_code/single_diag_gaussian_no_viterbi/md-eval-v21.pl

Google [7], using the speaker embeddings extracted by VGG speaker recognition model based on GhostVLAD [62]. The recurrent neural networks (RNN) are used in UIS-RNN algorithm while the deep neural networks (DNN) are used in VGG model, which is trained by the VoxCeleb dataset [63] of speaker utterances. The implemented UIS-RNN system is an offline system and the DER will be examined using the same tested data as our proposed offline and real-time system.

4.2.1. DER of Offline System

The DER and its breakdown (Missed, False Alarm and Speaker Error) of the proposed offline system on SAIVT-BNews database is illustrated on Table 2.

Missed	False Alarm	Speaker Error	DER (=Missed + False Alarm + Speaker Error)
0.7%	1.8%	13.0%	15.5%

Table 2 DER and its breakdown of proposed offline system

It can be shown that the DER of proposed offline system is 15.5% which is mainly from the speaker error (13%), on the tested data.

In order to compare the DER on audios of different number of speakers, evaluations are made on the audio files in turn based on the total number of speakers. The results are illustrated in Figure 16. The DER and its breakdown of the audio files with 1, 2, 3, 4 or more than 5 speakers, are shown respectively. The DER of audio files with any speakers (i.e. all tested data) is shown on the last column. The DERs of each type of files are shown in number of percentages on the top of each column.

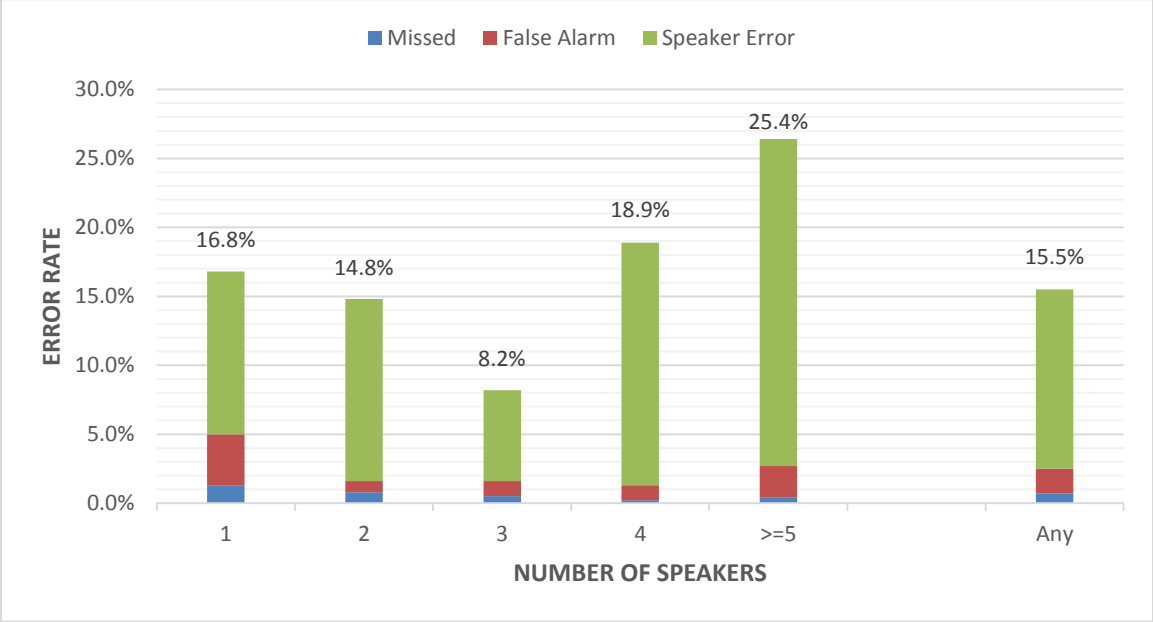

Figure 16 DER of audios with different number of speakers

The lowest DER 8.2% is achieved for audios with 3 speakers while the largest DER 26.4% is obtained for audios with more than 5 speakers, indicating the poor performance of the proposed offline system for situations when there are many (more than 5) speakers are speaking.

To compare the performance of proposed offline system and the UIS-RNN system, the DER of two systems for audios with different number of speakers are shown in Figure 17. The DERs of audio files with any speakers (i.e. all tested data) are shown on the last column. The DERs of each type of files are shown in number of percentages on the top of each column.

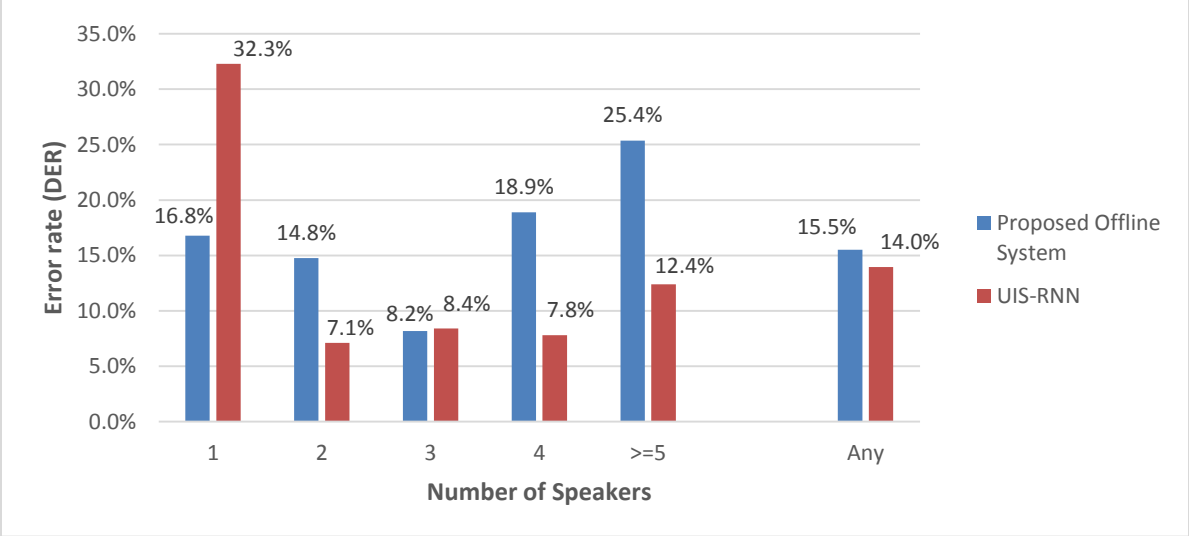

Figure 17 DER of proposed offline systems and UIS-RNN system

The UIS-RNN system has generally better accuracy except for the audio with only 1 speaker and it has better robustness for audios with more than 1 speakers compared with out proposed system. Considering the UIS-RNN system requires large amount of training data and long processing time, the overall DER difference (15.5% compared with 14.0%) of our proposed offline system and UIS-RNN system, is acceptable.

4.2.2. DER of Real-time System

In order to examine the DER of the real-time system, the snapshot of the diarization results will be made for every 30 second. The same snapshot will be taken on the reference ground true data, so that the DER can be evaluated pairwise. The results are illustrated on Figure 18. The DERs are shown in number of percentages on the top of each column.

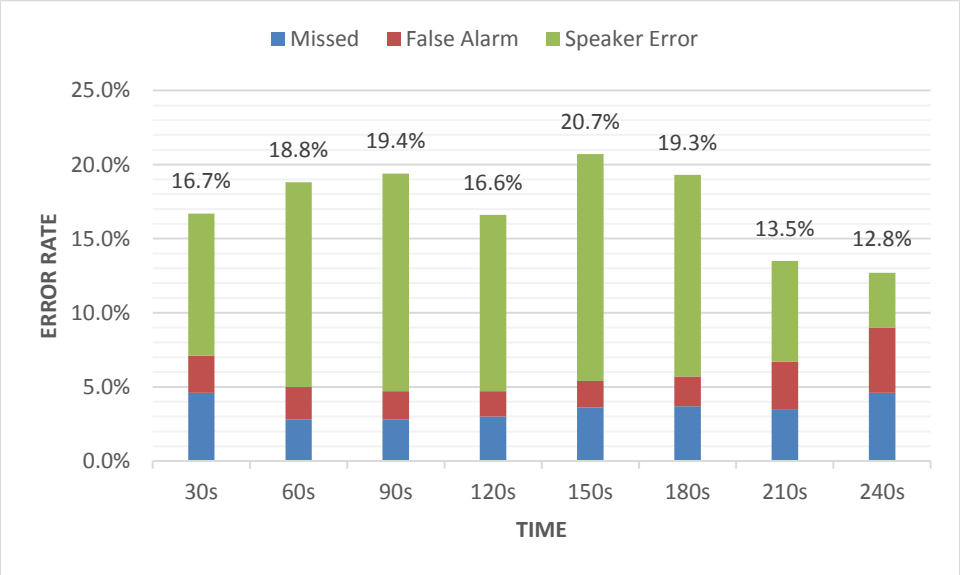

Figure 18 DER of proposed real-time system for every 30s

The DER of real-time system for different snapshot is generally stable over time and is approximately between 10% and 20%. The real-time system has better performance after 210s compared with that before 210s.

To compare the accuracy of the proposed real-time system with offline system and UIS-RNN system, the 30s snapshots of the tested audio data are processed by the proposed offline system and offline UIS-RNN system. The results of DER of three systems are illustrated in Figure 19.

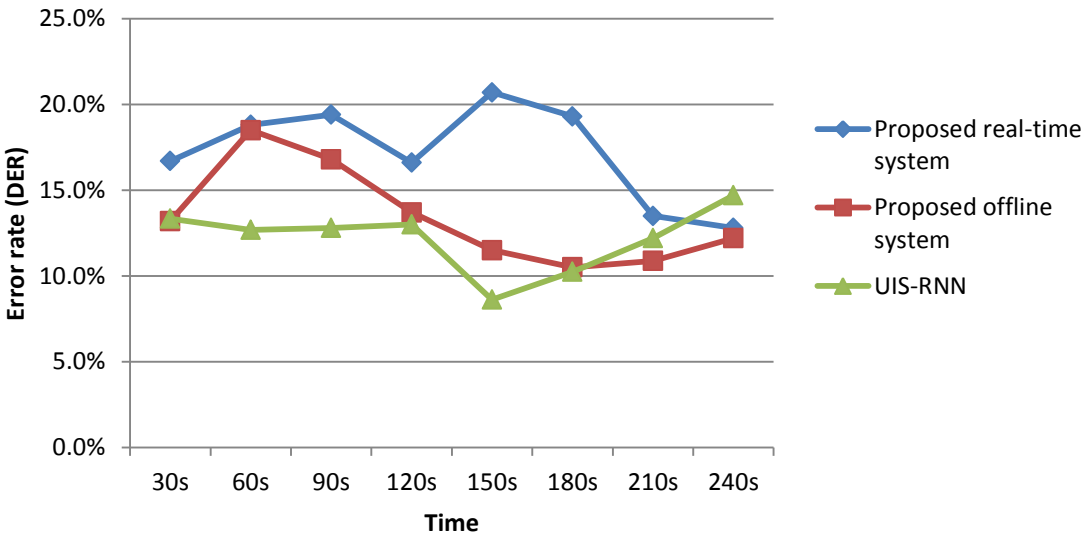

Figure 19 DER of proposed real-time, offline system and UIS-RNN system for every 30s

The real-time system has the largest DER while the UIS-RNN has the lowest DER. For each time snapshot, the DER of proposed offline system is lower than the proposed real-time system, which is reasonable as same core algorithms are used for two systems. The larger DER of real-time system compared with the offline system is resulted from the fact that in the real-time system, the KBM Training and offline clustering may not be completed within 1 second. The delay of KBM training and offline clustering will be further examined in the Section 4.3.2 and 4.3.3. The average DER of real-time system for every 30s snapshot from 30s to 240s is 17.2%, which is 3.8% higher than proposed

offline system and 5.0% higher than UIS-RNN system. Considering the fact that there is no time constraint to perform diarization for proposed offline and UIS-RNN system and the real-time system is required to generates results within 1s for every 1s new data, this difference of DER is still acceptable.

4.3. Other Experiments

In this Section, another three experiments that are mainly focusing on the elapsed time of the components of the proposed systems are conducted. The elapsed time analysis of the offline system are made in the Section 4.3.1 and KBM delay and online seconds analysis of the real-time system are made in Section 4.3.2 and Section 4.3.3.

4.3.1. Experiment on Offline System: Elapsed Time Analysis

The purpose of this experiment is to examine the speed of the overall offline system and each component.

The proposed speaker diarization system are used to processed multiple recorded audio files with duration ranging from 1 min to 50 min of every 1 min increment to see the elapsed time of each component and the total system. The corresponding real-time factor (xRT, defined by elapsed time / audio duration) is calculated. For each file, the system is

run for five times to take the average processing time to avoid the outliers. The resulted real-time factor for audios from 1 min to 50 min is plotted in Figure 20.

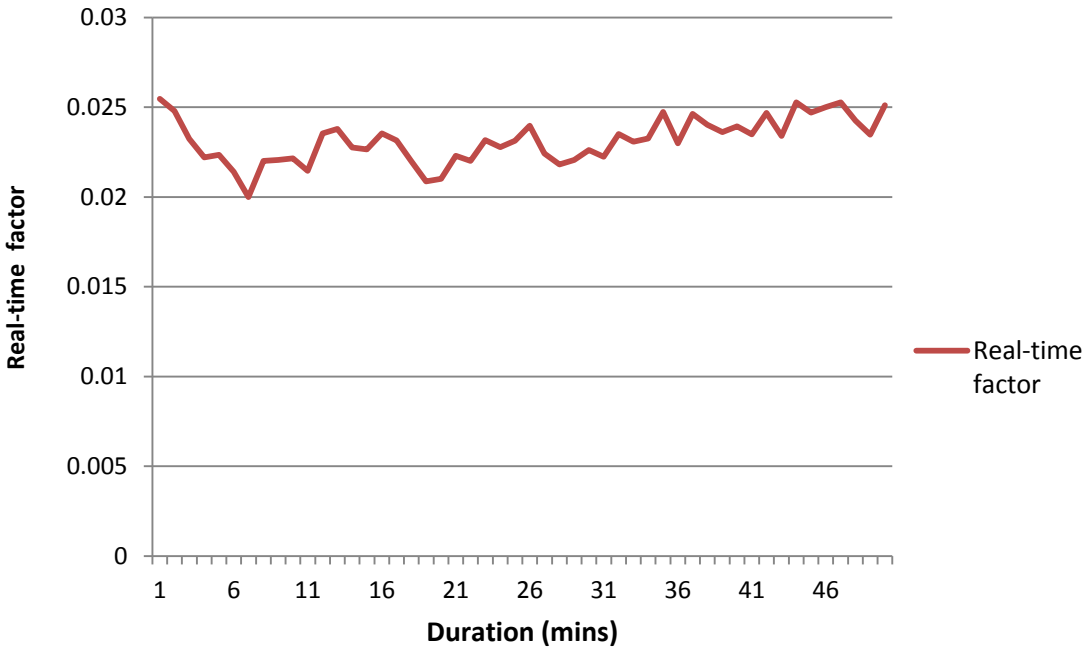

Figure 20 Offline system's real-time factor versus audio duration

The real-time factor is ranging between 0.02 and 0.025, demonstrating the high speed of the whole offline system for audios of different duration. The real-time factor is slowly increasing from audio of 7 min to audio of 50 min.

The average real-time factor of each component and the elapsed time for performing diarization on a 10 min audio are recorded as well and are summarized in Table 3.

Components	Average real-time factor for audio files with duration from 1 min to 50 min	Elapsed time of 10 min audio
VAD	0.0013	0.750s
Feature extraction	0.0015	1.016s
Segmentation	< 1e-5	0.016s
KBM Training	0.0014	0.622s
Compute representation (CV) for all features	0.0130	7.614s
Offline Clustering	0.0015	0.578s
Resegmentation	0.0046	2.700s
Total	**0.023**	**13.30s**

Table 3 Real-time factor of different components in proposed offline system

The average real-time factor of the total system on audio files is 0.023. The average running time for the whole system is 13.3s for 10 min (0.022xRT), which means that if the offline system is used for real-time audio stream of 10 min duration, the result is generated with more than 10 seconds' delay. Therefore, although the speed of the offline system is high, it is not realistic to use the offline system in the real-time situation when the data duration keeps increasing. Some components in the system need to be conducted

in the online manners that only newly input data is processed. For components in the system that has to be conducted in offline manners, separate threads need to be used, which supports our design of the real-time system in Section 3.2.

The VAD, features extraction and segmentation are running really fast and achieve low real-time factors. This means in real-time application, when these components performed in online manner, these processes can generate the output for the newly input data with short delay.

The most time-consuming part is the KBM training and corresponding CVs computation for all features. The average real-time factor is 0.0014 for KBM training and 0.0130 for CVs computation for all features. For a 10min audio, the elapsed time is 0.622s for KBM training and 7.614s for CVs computation. Therefore this part is necessarily run in a separate thread, as proposed in Section 3.2.

The AHC achieve average real-time factor of 0.0015. The average running time for the KBM training is 0.578s for 10min audio (0.00096 xRT) and 0.953s for 15min audio (0.00106 xRT). This means that the AHC can generate diarization results within 1s for audio with duration lower than 15min. This speed of the AHC method demonstrates that the offline clustering result can be used to constantly update the online clustering results in the proposed real-time system.

4.3.2. Experiment on Real-time System: KBM Delay Analysis

The results of the elapsed time analysis of the proposed offline system in Section 4.3.1 shows that the process of KBM training may require a few seconds when the audio duration is long, which means that the KBM model may not be updated for every new data/observation in the proposed real-time system. Therefore at time t_2, the KBM that used to compute CV of feature vectors for the later clustering process is trained by the data up to time t_1 , where $t_1 \leq t_2$ and we define the $\boldsymbol{KBM\ Delay := t_2 - t_1}$. As the KBM is used to represent the acoustic space of the input audio, the KBM delay can be considered to represent how many seconds of the audio data that is not reflective in the KBM. It should be noted that the KBM delay is not the delay of the diarization results, but only the delay of the current speaker model in representing the acoustic space of the audio data. When KBM delay = 0, the current KBM covers the acoustic space of all data of the input audio. When KBM delay = 10, the latest 10 second of the audio data is not included in the training of the current KBM, that means if there is new speaker shown in this 10 second of audio, the current KBM will fail to capture the unique acoustic characteristics of this unique new speaker. Therefore, the larger KBM delay indicates higher delay in detecting new speaker.

As the KBM training is an offline process, that means in our offline system, the KBM delay is always 0 because the KBM training process is performed when all audio data is available. However in the real-time system, the KBM training process is kept running for new data in the separate thread, so the KBM Delay will be larger than 0 and will be increasing as the data size is increasing.

To examine the relations between KBM delay and audio duration, an experiment is designed to record the average KBM delay every 10 second for the real-time system on a 20-minute audio stream. The result is illustrated in Figure 21.

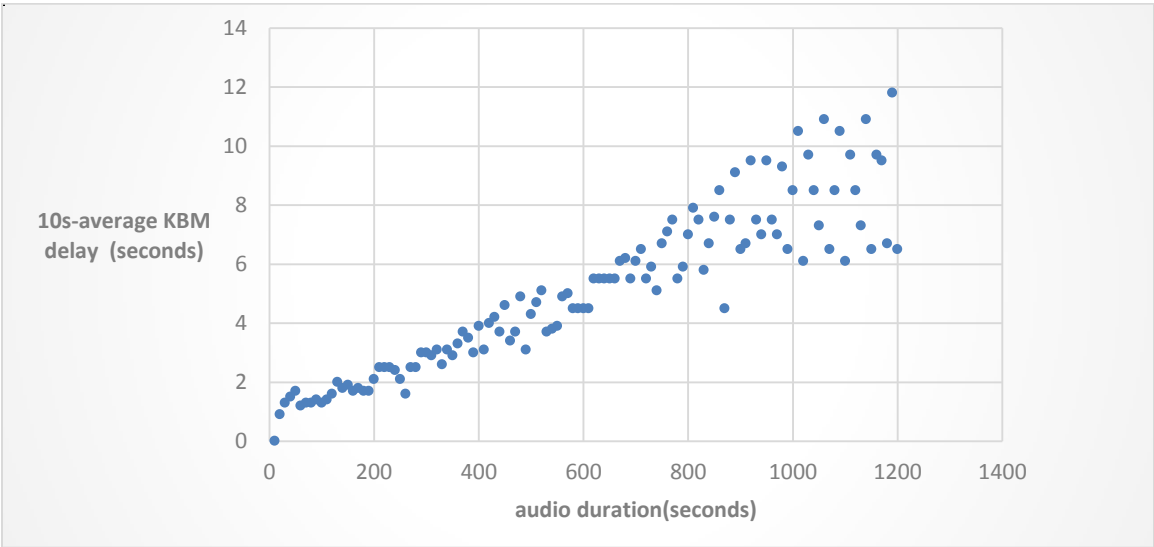

Figure 21 KBM delay versus audio duration

It is shown that the KBM delay is increasing as the audio duration is increasing, which is reasonable as the KBM training is performed on all the audio data available. The average KBM delay is below 6 seconds for audio below 10 minutes and below 12 seconds for audio below 20 minutes.

Please note that the KBM delay is not the delay of the result generating. The results will still be generated by the real-time system within 1s for 1s new data. The KBM delay is the model delay to represent the current data.

4.3.3. Experiment on Real-time System: Online Seconds Analysis

The clustering step of the real-time system are divided into two components, the offline clustering that conducted using a separate thread and the simple online clustering that is used when the offline clustering is not completed for all data available. Therefore, the final speaker labels consist of results from offline clustering and results from online clustering.

Assume at time t_2, the offline clustering just completed for data up to time t_1 ($t_1 \leq t_2$), the online clustering need to be conducted for data between t_1 and t_2 such that the all data up to t_2 can be labelled. As illustrated in Figure 22, the segments up to t_1 is labelled by offline clustering and the segments from t_1 to t_2 are labelled by online clustering.

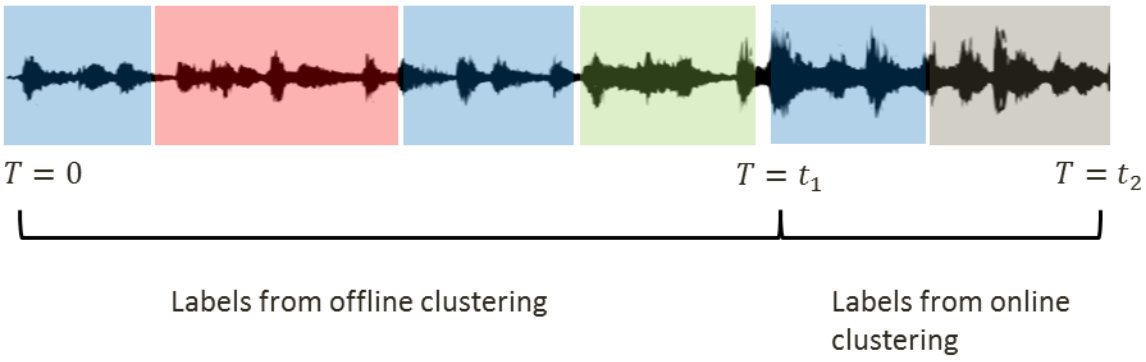

Figure 22 Speaker lables from offline and online clustering in the real-time system

Because in our experiment, the real-time data will be simulated using recoded audio such that the data is fed into the system every one second, therefore, t_1 and t_2 are in unit of second and the value of $t_2 - t_1$ is an integer. We define $\boldsymbol{Online\ Seconds := t_2 - t_1}$ which represent the number of seconds in the audio data that are labelled by online clustering.

As the performance of online clustering is worse than offline clustering, it is necessary to examine the relationship between audio duration and online seconds to see the proportion of final results that are from online clustering. An experiment is designed to record the average online seconds every 10 second for the real-time system on a 20-minute audio stream. The result is illustrated in Figure 23.

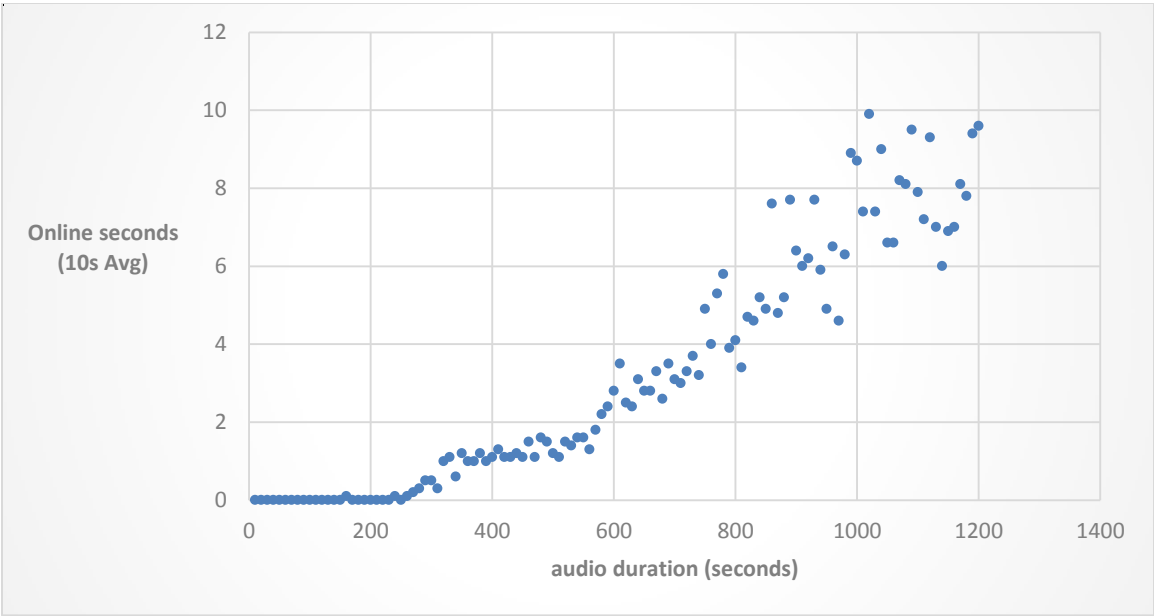

Figure 23 Online seconds versus audio duration

It is shown that for audio with duration below 250s, the offline clustering can be generally completed within 1s so all speaker labels are from offline clustering and online seconds equals 0. As the audio duration increase, the online seconds increase as the time to perform offline system is increasing for more data/segments are available. For audio below 1200s (20mins), the online seconds are below 10s, which means that the majority speaker labels are from the offline clustering and the online clustering will only impose minor impact on the overall performance of the real-time system.

4.4. Other Observations

There are other two observations made during the testing of the proposed diarization system.

Firstly, the proposed system fails to distinguish between music and speech and usually classify the music in the broadcast news as one speaker. For example, the speaker 1 (marked in blue color) in the diarization of the example illustrated in Figure 24, are from the music in the start and end of the news.

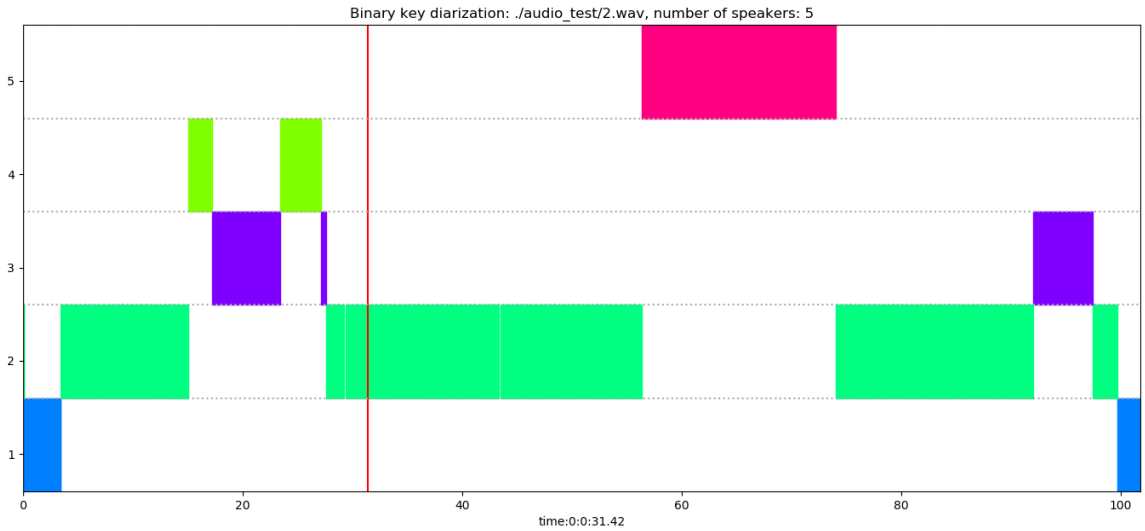

Figure 24 Diarization result of the broadcast news with ID 3054300 in AIVT-BNews database

This problem of failure to remove music is because we only use a VAD that separates voice and silence, so music will also be included in the consequent steps of feature extraction and clustering. This problem will increase the overall DER (mainly the false

alarm error) but won’t cause large confusion and large negative effect on the usability of the proposed system, as the music can still be distinguished from the speech of the speakers.

Secondly, the diarization has also been tested on several audio where the languages is Chinese and the no evidence is shown that the system have worse accuracy for Chinese language audio. Theoretically, the system is language-independent because the acoustic features (MFCCs) that we used depends no language-specific information.

5. Conclusions

This dissertation proposes a speaker diarization system that incorporates the binary key modelling approaches for offline and real-time speaker diarization tasks. The proposed system is domain-robust, operating-system free, language-independent and requires no external training data. A visualization panel is also designed for the users of the system to examine the diarization results.

The real-time version of the system is proposed on the basis of the offline system, which is the main contribution of this dissertation to fill the gap of the limited research on the real-time diarization system. Although it is stated in some literatures that only online diarization techniques can be used for real-time application, this dissertation proposes to combine components that operates in online manner including VAD, feature extraction and online clustering, with components that operates in offline manner including KBM training, CV computation for features and offline AHC clustering. Consequently, the results generated by the real-time system can be corrected constantly by the results of offline system, thanks to high processing speed of the system.

The DER is evaluated on the proposed offline system, real-time system and the current state-of-art UIS-RNN offline system. The DER of the proposed offline system is 15.5% on tested files, 1.5% higher than UIS-RNN system. The average DER of the proposed real-time system for every 30s snapshot is 3.8% higher than the proposed offline system and 5.0% higher than offline UIS-RNN system. The elapsed time analysis of each

component in the system is made to demonstrate the speed of the overall system, and the low time delay for the real-time application.

There are several drawbacks of the proposed system and questions that remain answers in this dissertation and therefore the directions of the further investigations are suggested as follows:

- It is observed that the number of speakers change frequently in the real-time system for some audios. This problem needs to be further analyzed and solved for the better usability of the real-time system.

- The parameters used to run the experiment of the proposed real-time system are from the pervious works of offline system. The relationship between parameters and DER need to be further analyzed in the real-time situation to determine the better parameter sets for real-time system.

- The binary key speaker model that used in the proposed system performs model training in an offline fashion that includes all data available. Although the training process is fast, it still leads to the delay in model training in real-time system when there is large amount of audio data. It is possible to revise the current algorithm to store the previous models and train only the new data so that the model can be incrementally update. This is worth investigating in the further research.

- The data used to test the proposed system is limited because many datasets used by the other research are not free. The performance of the proposed system need to be further examined using other dataset.

- The real-time system is only tested using the recorded audios that is simulated as the real-time data. The system needs to be further tested or improved in terms of speed and audio processing techniques for the actual real-time streaming data.

References

[1] D. A. Reynolds and P. Torres-Carrasquillo, "Approaches and Applications of Audio Diarization," *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05),* vol. V, pp. 953-956, 2005.

[2] X. A. Miro, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland and O. Vinyals, "Speaker Diarization: A Review of Recent Research," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 20, no. 2, pp. 356-370, 2012.

[3] S. E. Tranter and D. A. Reynolds, "An overview of automatic speaker diarization systems," *IEEE Transactions on audio, speech, and language processing,* vol. 14, 2006.

[4] X. Anguera Miró, "Robust speaker diarization for meetings," *Universitat Politècnica de Catalunya.,* 2006.

[5] X. Anguera, C. Wooters and J. Hernando, "Acoustic Beamforming for Speaker Diarization of meeting," *IEEE Transactions on Audio, Speech, and Language Processing,* pp. 2011-2022, 2007.

[6] M. Kunešová, Z. Zajíc and V. Radová, "Experiments with Segmentation in an Online Speaker Diarization System," *International Conference on Text, Speech, and Dialogue*, pp. 429-437, 2017.

[7] A. Zhang, Q. Wang, Z. Zhu, J. Paisley and C. Wang, "Fully supervised speaker diarization," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[8] W. Zhu and J. Pelecanos, "Online speaker diarization using adapted i-vector transforms," *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5045-5049, 2016.

[9] S. Araki, M. Fujimoto, K. Ishizuka, H. Sawada and S. Makino, "A DOA based speaker diarization system for real meetings," *Hands-Free Speech Communication and Microphone Arrays*, pp. 29-32, 2008.

[10] J. Geiger, F. Wallhoff and G. Rigoll, "GMM-UBM based open-set online speaker diarization," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[11] K. Otsuka, S. Araki, K. Ishizuka, M. Fujimoto, M. Heinrich and J. Yamato, "A realtime multimodal system for analyzing group meetings by combining face pose

tracking and speaker diarization," in *Proceedings of the 10th international conference on Multimodal interfaces*, 2008.

[12] G. Garau and H. Bourlard, "Using audio and visual cues for speaker diarisation initialisation," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2010.

[13] E. Gonina, G. Friedland, H. Cook and K. Keutzer, "Fast Speaker Diarization Using a High-LevelScripting Language," in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, 2011.

[14] D. Dimitriadis and P. Fousek, "Developing On-Line Speaker Diarization System," in *INTERSPEECH*, Stockholm, Sweden, 2017.

[15] T. H. Nguyen, E. S. Chng and H. Li, "Speaker Diarization: An Emerging Research," in *Speech and Audio Processing for Coding, Enhancement and Recognition*, New York, Springer, 2015, pp. 229-277.

[16] J. Patino, R. Y. Héctor Delgado, H. Bredin, C. Barras and N. Evan, "ODESSA at Albayzin Speaker Diarization Challenge 2018," in *IberSPEECH*, Barcelona, Spain, 2018.

[17] G. Friedland, A. Janin, D. Imseng, X. A. Miro, L. Gottlieb, M. Huijbregts, M. Knox and O. Vinyals, "The ICSI RT-09 speaker diarization system," in *IEEE Transactions on Audio, Speech, and Language Processing*, 2011.

[18] X. Anguera, C. Wooters, B. Peskin and M. Aguiló, "Robust Speaker Segmentation for Meetings: The ICSI-SRI Spring 2005 Diarization System," *International Workshop on Machine Learning for Multimodal Interaction,* pp. 402-414, 2005.

[19] Barras, Claude, X. Zhu, S. Meignier and J.-L. Gauvain, "Multistage speaker diarization of broadcast news," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1502-1512, 2006.

[20] D. Istrate, C. Fredouille, S. Meignier, L. Besacier and J. Bonastre, "NIST RT'05S evaluation: pre-processing techniques and speaker diarization on multiple microphone meetings," in *International Workshop on Machine Learning for Multimodal Interaction*, Springer, Berlin, 2005, pp. 428-439.

[21] D. A. v. Leeuwen, "The TNO speaker diarization system for NIST RT05s meeting data," in *International Workshop on Machine Learning for Multimodal Interaction*, Springer, Berlin, 2005, pp. 440-449.

[22] D. Liu and F. Kubala, "Fast speaker change detection for broadcast news

transcription and indexing," *Sixth European Conference on Speech Communication and Technology,* 1999.

[23] Davis, Steven and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE transactions on acoustics, speech, and signal processing,* vol. 28, no. 4, pp. 357-366, 1980.

[24] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *the Journal of the Acoustical Society of America,* vol. 87, no. 4, pp. 1738-1752, 1990.

[25] P. S. Jadhav, "Classification of Musical Instruments sounds by Using MFCC and Timbral Audio Descriptors," *International Journal on Recent and Innovation Trends in Computing and Communication,* vol. 3, no. 7, pp. 52-56, 2017.

[26] H. Delgado, M. Todisco, M. Sahidullah, A. K. Sarkar, N. Evans, T. Kinnunen and Z.-H. Tan, "Further Optimisations of Constant Q Cepstral Processing for Integrated Utterance and Text-dependent Speaker Verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, 2016.

[27] D. A. Reynolds, "Experimental evaluation of features for robust speaker identification," *IEEE Transactions on Speech and Audio Processing,* pp. 639-643,

1994.

[28] R. Pawar, P.P.Kajave and S.N.Mali, "Speaker Identification using Neural Networks," in *Proceeding of world Academy of Science, Engineering and Technology*, Prague, 2005.

[29] V. Tiwari, "MFCC and its applications in speaker recognition," *International journal on emerging technologies,* vol. 1, no. 1, pp. 19-22, 2010.

[30] G. Schwarz, "Estimating the dimension of a model," *The annals of statistics,* vol. 6(2), pp. 461-464, 1978.

[31] S. S. Chen and P. Gopalakrishnan, "Speaker, environment and channel change detection and clustering via the bayesian information criterion," *Proc. DARPA broadcast news transcription and understanding workshop,* vol. 8, pp. 127-132, 1998.

[32] M. Rouvier, G. Dupuy, P. Gay, E. Khoury, T. Merlin and S. Meignier, "An Open-source State-of-the-art Toolbox for Broadcast News Diarization," *Interspeech,* 2013.

[33] Y. Huang, O. Vinyals, G. Friedland, C. Müller, N. Mirghafori and C. Wooters, "A fast-match approach for robust, faster than real-time speaker diarization," *2007 IEEE Workshop on Automatic Speech Recognition & Understanding (ASRU),* pp. 693-698,

2008.

[34] H. Gish, M. H. Siu and R. Rohlicek, "Segregation of speakers for speech recognition and speaker identification," *1991 International Conference on Acoustics, Speech, and Signal Processing,* pp. 873-876, 1991.

[35] M. A. Siegler, U. Jain, B. Raj and R. M. Stern, "Automatic segmentation, classification and clustering of broadcast news audio," *Proc. DARPA speech recognition workshop,* 1997.

[36] R. Wang, M. Gu, L. Li, M. Xu and T. F. Zheng, "Speaker segmentation using deep speaker vectors for fast speaker change scenarios," *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* pp. 5420-5424, 2017.

[37] F. Castaldo, D. Colibro, E. Dalmasso, P. Laface and C. Vair, "Stream-based speaker segmentation using speaker factors and eigenvoices," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008.

[38] L. Wan, Q. Wang, A. Papir and I. L. Moreno, "Generalized end-to-end loss for speaker verification," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP),* 2018.

[39] D. Garcia-Romero, D. Snyder, G. Sell, D. Povey and A. McCree, "Speaker diarization using deep neural network embeddings," *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4930-4934, 2017.

[40] G. Sell and D. Garcia-Romero, "Speaker diarization with plda i-vector scoring and unsupervised calibration," *2014 IEEE Spoken Language Technology Workshop (SLT)*, pp. 413-417, 2014.

[41] D. Liu and F. Kubala, "Online speaker clustering," *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing - ICASSP '03*, vol. 1, 2003.

[42] P. Kenny, D. Reynolds and F. Castaldo, "Diarization of Telephone Conversations using Factor Analysis," *IEEE Journal of Special Topics in Signal Processing*, vol. 4, no. 6, pp. 1059-1070, 2010.

[43] D. Reynolds, P. Kenny and F. Castaldo, "A Study of New Approaches to Speaker Diarization," in *Tenth Annual Conference of the International Speech Communication Association*, Brighton, 2009.

[44] S. Shum, N. Dehak, E. Chuangsuwanich, D. Reynolds and J. Glass, "Exploiting intra-conversation variability for speaker diarization," in *Twelfth Annual Conference*

of the International Speech Communication Association, 2011.

[45] G. Sell and D. Garcia-Romero, "Diarization resegmentation in the factor analysis subspace," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[46] NIST., "The 2009 (RT-09) Rich Transcription Meeting Recognition Evaluation Plan," 2009.

[47] X. Anguera and J.-F. Bonastre, "Fast speaker diarization based on binary keys," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011.

[48] J. Patino, H. Delgado, N. Evans and X. Anguera, "EURECOM submission to the Albayzin 2016 speaker diarization evaluation," *Proc. IberSPEECH*, 2016.

[49] "WebRTC," [Online]. Available: https://webrtc.org/. [Accessed 2019].

[50] J. H. Ko, J. Fromm, M. Philipose, I. Tashev and S. Zarar, "Limiting Numerical Precision of Neural Networks to Achieve Real-Time Voice Activity Detection," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Calgary, AB, 2018.

[51] "py-webrtcvad," 2019. [Online]. Available: https://github.com/wiseman/py-webrtcvad. [Accessed June 2019].

[52] B. McFee, C. Raffe, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg and O. Nieto, "librosa: Audio and Music Signal Analysis in Python," in *Proceedings of the 14th python in science conference*, 2015.

[53] "Librosa," 2018. [Online]. Available: https://librosa.github.io/librosa/. [Accessed Jun 2019].

[54] X. Anguera and J.-F. Bonastre, "A Novel Speaker Binary Key Derived from Anchor Models," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[55] "SciPy," 2019. [Online]. Available: https://www.scipy.org/. [Accessed Jun 2019].

[56] G. Hernandez-Sierra, J. R.Calvo, J.-F. Bonastre and P.-M. Bousquet, "Session compensation using binary speech representation for speaker recognition," *Pattern Recognition Letters*, pp. 17-23, 2014.

[57] X. Anguera, C. Wooters and J. M. Pardo, "Robust Speaker Diarization for Meetings: ICSI RT06s evaluation system," in *Ninth International Conference on Spoken*

Language Processing, 2006.

[58] J. Rougui, M. Rziza, D. Aboutajdine, M. Gelgon and J. Martinez, "Fast incremental clustering of gaussian mixture speaker models for scaling up retrieval in on-line broadcast," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, 2006.

[59] T. H. Nguyen, E. S. Chng and H. Li, "T-test distance and clustering criterion for speaker diarization," in *Ninth Annual Conference of the International Speech Communication Association*, 2008.

[60] H. Ghaemmaghami, D. Dean and S. Sridharan, "Speaker attribution of Australian broadcast news data," in *Proceedings of the First Workshop on Speech, Language and Audio in Multimedia (SLAM): CEUR Workshop Proceedings*, 2013.

[61] ICSI Meetings Recorder corpus, 2006. [Online]. Available: http://www1.icsi.berkeley.edu/Speech/mr/.

[62] W. Xie, A. Nagrani, J. S. Chung and A. Zisserman, "Utterance-level Aggregation for Speaker Recognition in the Wild," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[63] A. Nagrani, J. S. Chung and A. Zisserman, "VoxCeleb: a large-scale speaker identification dataset," in *INTERSPEECH*, 2017.